Oracle® AI Database

Command Line Interface Reference

26ai
G49015-02
May 2026

Oracle AI Database Command Line Interface Reference, 26ai

G49015-02

Primary Author: Preeti Shukla

Contributing Authors: Priyanka Deondi

Contents

Preface

The *Oracle GoldenGate Microservices Architecture Documentation* contains the Oracle GoldenGate concepts, tasks, advance tasks, security, and other reference information.

Audience

This guide is intended for developers, database users, and administrators who are responsible for installing, deploying, and configuring Oracle GoldenGate. It is assumed that readers are familiar with web technologies, low code development, and scripting. A general understanding of Windows and UNIX platforms is required.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customer access to and use of Oracle support services will be pursuant to the terms and conditions specified in their Oracle order for the applicable services.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, such as "From the File menu, select **Save**." Boldface also is used for terms defined in text or in the glossary.
italic `*italic*`	Italic type indicates placeholder variables for which you supply particular values, such as in the parameter statement: `TABLE *table_name*`. Italic type also is used for book titles and emphasis.
`monospace` `MONOSPACE`	Monospace type indicates code components such as user exits and scripts; the names of files and database objects; URL paths; and input and output text that appears on the screen. Uppercase monospace type is generally used to represent the names of Oracle GoldenGate parameters, commands, and user-configurable functions, as well as SQL commands and keywords.
UPPERCASE	Uppercase in the regular text font indicates the name of a process or utility unless the name is intended to be a specific case. Keywords in upper case (`ADD EXTRACT`, `ADD EXTTRAIL`, `FORMAT RELEASE`).
LOWERCASE	Names of processes to be written in lower case. Examples: `ADD EXTRACT exte`, `ADD EXTRAIL ea`.

Convention	Meaning
{ }	Braces within syntax enclose a set of options that are separated by pipe symbols, one of which must be selected, for example: {*option1* | *option2* | *option3*}.
[ ]	Brackets within syntax indicate an optional element. For example in this syntax, the SAVE clause is optional: CLEANUP REPLICAT *group_name* [, SAVE *count*]. Multiple options within an optional element are separated by a pipe symbol, for example: [*option1* | *option2*].
Sample Locations	Compass directions such as east, west, north, south to be used for demonstrating Extract and Replicat locations. Datacenters names to use the standard similar to dc1, dc2.
Group names	Prefixes for each process, as follows: • Extract: ext. Usage with location: extn, where ***n*** indicates 'north' compass direction. • Replicat: rep. Usage with location: repn, where ***n*** indicates 'north' compass direction. • Distribution Path: dp. Usage with location: dpn, where ***n*** indicates 'north' compass direction. • Checkpoint table: ggs_checkpointtable • Trail file names: e or d depending on whether the trail file is for the Extract or distribution path. Suffix derived in alphabetical order. Usage for an Extract trail file: ea, eb, ec. • Trail file subdirectory: The name will use compass directions to refer to the trail subdirectories. Example for trail subdirectory name would be /east, /west, /north, /south.

Related Information

Oracle GoldenGate Documentation

Oracle GoldenGate for Distributed Applications and Analytics

Oracle GoldenGate Studio Documentation

OCI GoldenGate

Oracle Database High Availability

Oracle GoldenGate Veridata

1

About Admin Client Command Line Interface

Admin Client is the command line interface (CLI) for Oracle GoldenGate Microservices Architecture (MA).

What is the Admin Client?

Admin Client is a command line utility that can be used to create, modify, and remove Oracle GoldenGate processes and can be used in place of the MA web user interface. The Admin Client program is located in the `$OGG_HOME/bin` directory, where `$OGG_HOME` is the Oracle GoldenGate home directory. The name of the Admin Client program is `adminclient`.

Also see Microservices Command Line Interface in the *Oracle GoldenGate Microservices Documentation*.

2

Admin Client Command Line Interface Commands

Learn how to use the Oracle GoldenGate Microservices Architecture Admin Client commands and options, and review examples.

!

Use the `!` command to execute a previous Admin Client command without modifications. To modify a command before executing it again, use the `FC` command. To display a list of previous commands, use the `HISTORY` command. The `!` command without arguments executes the most recent command. Options enable you to execute any previous command by specifying its line number or a text substring. Previous commands can be executed again only if they were issued during the current session, because command history is not maintained from session to session.

Syntax

```
! n
```

n
Executes the command from the specified line. Each command line is sequenced, beginning with 1 at the start of the session.

Examples

```
! 9

! -3

! sta
```

ADD AUTHORIZATIONPROFILE

When a profile is created for an Oracle GoldenGate deployment, the content which describes the profile will exist only in that deployment. This applies for both Service Manager deployments and non-Service Manager deployments. Information that is not sensitive will be stored with other configuration data. All profile information will be available throughout the entire Oracle GoldenGate deployment however, the profile information is not shared across deployments.

Only security administrators for an Oracle GoldenGate deployment can create authorization profiles. Security administrators can only create an authorization profile, which holds information for a specific IDP server and application.

> **Note**
>
> You cannot create, modify or delete the `localCredentialStore` profile. This profile will always exist.

Syntax

```
ADD AUTHORIZATIONPROFILE profile-name
    DEPLOYMENT deployment-name
    IDCS
    ID client-id [ SECRET client-secret ]
    DISCOVERYURI discovery-uri
    GROUPS
        SECURITY security-group
        [ ADMINISTRATOR administrator-group ]
        [ OPERATOR operator-group ]
        [ USER user-group ]
    [ TTLSECONDS ttl-number ]
    [ DESCRIPTION description ]
```

profile-name
Name of the authorization profile.

deployment-name
Name of the deployment associated with the authorization profile.

ID *client-id* SECRET *client-secret*
Specify the IDP Application's client ID and IDP Application's Client Secret (securely stored).

DISCOVERYURI *discovery-uri*
IDP server's OpenID Discovery Docs endpoint.

GROUPS
IDP groups to Oracle GoldenGate user roles mapping. Possible values are:

SECURITYGROUP *security-group* (Mandatory)

[ADMINGROUP *admin-group*]

[OPERATORGROUP *operator-group*]

[USERGROUP *user-group*]

See Add New Users to the Deployment to know about Oracle GoldenGate user roles and privileges.

TTL *value*
The time, in seconds, needed to pass before the OpenID JSON Web Key (JWK) containing the OpenID signing certificate used to validate an access token needs to be queried again.

DESCRIPTION
Describe the authorization profile.

Example

Here's an example of adding an authorization profile with 2 mapped groups:

```
ADD AUTHORIZATIONPROFILE apn
DEPLOYMENT IDCS CLIENT ID SECRET DISCOVERYURI
GROUPS SECURITY group_security OPERATOR group_operator
```

> **Note**
>
> When you successfully create the authorization profile, the system will not show any success message. This behavior occurs with other commands use for authorization profile management, including ALTER AUTHORIZATIONPROFILE, VALIDATE AUTHORIZATIONPROFILE, and DELETE AUTHORIZATIONPROFILE.

ADD CHECKPOINTTABLE

Not valid for Replicat for Java, Oracle GoldenGate Applications Adapter, or Oracle GoldenGate for Big Data.

Valid for non-parallel Replicats.

For a parallel Replicat, the checkpoint table is created as part of the command to add the parallel Replicat and does not require that the checkpoint table be created before adding the parallel Replicat.

Use `ADD CHECKPOINTTABLE` to create a checkpoint table in the target database. Replicat uses the table to maintain a record of its read position in the trail for recovery purposes.

The use of a checkpoint table is strongly recommended, because it causes checkpoints to be part of the Replicat transaction. This allows Replicat to recover more easily in certain circumstances than when a checkpoint file alone is used. Parallel and coordinated Replicats require checkpoint tables.

One table can serve as the default checkpoint table for all Replicat groups in an Oracle GoldenGate instance if you specify it with the `CHECKPOINTTABLE` parameter in a `GLOBALS` file. More than one instance of Oracle GoldenGate (multiple installations) can use the same checkpoint table. Oracle GoldenGate keeps track of the checkpoints even when the same Replicat group name exists in different instances.

Use the `DBLOGIN` command to establish a database connection before using this command. Do not change the names or attributes of the columns in this table. You may, however, change table storage attributes.

See Add a Checkpoint Table.

Syntax

```
ADD CHECKPOINTTABLE [[container. | catalog.] owner.table]
```

The name cannot contain any special characters, such as quotes, backslash, dollar sign, and percent symbol. Record the name of the table, because you need it to view statistics or delete the table if needed.

container. | catalog.
The Oracle pluggable database, if applicable. If this option is omitted, the pluggable database defaults to the one that is associated with the SOURCEDB, USERID, or USERIDALIAS portion of the DBLOGIN command (depending on the database).

owner.table
The owner and name of the checkpoint table to be created. The name cannot contain any special characters, such as quotes, backslash, dollar sign, and percent symbol.
The name of a MySQL checkpoint table can contain no more than 58 characters.
The owner and name can be omitted if you are using this table as the default checkpoint table and it is listed with CHECKPOINTTABLE in the GLOBALS file.
It is recommended, but not required, that the table be created in a schema dedicated to Oracle GoldenGate. If an owner and name are not specified, a default table is created based on the CHECKPOINTTABLE parameter in the GLOBALS parameter file.
Record the name of the checkpoint table as that will be used when you add a Replicat, or delete a Replicat and need to drop the checkpoint table using the DELETE CHECKPOINTTABLE command.
In MA, the default schema for the checkpoint table is controlled by the Oracle GoldenGate user that is defined for each deployment. See Replication Settings.

Examples

The following adds a checkpoint table with the default name specified in the GLOBALS file.

```
ADD CHECKPOINTTABLE
```

The following adds a checkpoint table with a user-defined name.

```
ADD CHECKPOINTTABLE ggadmin.ggs_checkpoint
```

ADD CREDENTIALS

The ADD CREDENTIALS command adds a new username and password or access token to a secure store that resides on the same system where the Admin Client is running. This credential is used to log in to Oracle GoldenGate Service Manager and Admin Client command line using the CONNECT command.

The username and password are referenced using a credential name.

The ADD CREDENTIALS command also provides the access-token parameter that enables the Admin Client to connect to an IDP-enabled deployment.

Syntax

```
ADD CREDENTIALS credentials-name | USER username | [ PASSWORD password ] |
[TOKEN access-token]
```

Examples

```
ADD CREDENTIALS dbnorth USER ggadmin PASSWORD oggadmin-A1
```

In this example, a credential called `dbnorth` is added with a username `ggadmin` and password `oggadmin-A1`. If the password is not specified in the `ADD CREDENTIALS` command, you will be prompted to enter the password interactively, without displaying it on the terminal.

```
ADD CREDENTIALS dbnorth TOKEN
eyJ4NXQjUzI1NiI6Ii1ubWUzeFRCUUREbmZOblZLb3RrZ01ERUsxSmQ5X2dRMWo2RWlPRmJoeEkiLC
J4NXQiOiJUYjVvYXpFOURxbUE1ZXVwTXZuelNzdkJEaHciLCJraWQiOiJTSUdOSU5HX0tFWSIsImFs
ZyI6IlJTMjU2In0.eyJ1c2VyX3R6IjoiQW1lcmljYVwvQ2hpY2FnbyIsInN1YiI6Im9nZ19jb21tb2
4iLCJ1c2VyX2xvY2FsZSI6ImVuIiwic2lkbGUiOjQ4MCwiaWRwX25hbWUiOiJVc2VyTmFtZVBhc3N3
b3JkIiwidXNlci50ZW5hbnQubmFtZSI6ImlkY3MtMTdlY2EwMmQzMWE2NDVlNzg3MTBhZDUyYWMwMW
RkNDAiLCJpZHBfZ3VpZCI6IlVzZXJOYW1lUGFzc3dvcmQiLCJhbXIiOlsiVVNFUk5BTUVfUEFTU1dP
UkQiXSwiaXNzIjoiaHR0cHM6XC9cL2lkZW50aXR5Lm9yYWNsZWNsb3VkLmNvbVwvIiwidXNlcl90ZW
5hbnRuYW1lIjoiaWRjcy0xN2VjYTAyZDMxYTY0NWU3ODcxMGFkNTJhYzAxZGQ0MCIsImNsaWVudF9p
ZCI6InRva2VuR2VuZXJhdG9yIiwic3ViX3R5cGUiOiJ1c2VyIiwiY2xpZW50X3RlbmFudG5hbWUiOi
JpZGNzLW9yYWNsZSIsInJlZ2lvbl9uYW1lIjoidXMtYXNoYnVybi0xIiwidXNlcl9sYW5nIjoiZW4i
LCJleHAiOjE2ODY4NjI3NDAsImlhdCI6MTY4Njg2MjU2MCwiY2xpZW50X2d1aWQiOiJmZjY4ZGZjOT
Y5MGM0NTFiOTQ4NDU2YWY3NWQ3ZDM5YSIsImNsaWVudF9uYW1lIjoidG9rZW5HZW5lcmF0b3IiLCJp
ZHBfdHlwZSI6IkxPQ0FMIiwidGVuYW50IjoiaWRjcy0xN2VjYTAyZDMxYTY0NWU3ODcxMGFkNTJhYz
AxZGQ0MCIsImp0aSI6ImY5ZTVmYTE1YzJiYzQxOTBiNTFkMjRkNzk4YzBhMmU1IiwiZ3RwIjoiand0
IiwidXNlcl9kaXNwbGF5bmFtZSI6Im9nZyBjb21tb24iLCJvcGMiOmZhbHNlLCJzdWJfbWFwcGluZ2
F0dHIiOiJ1c2VyTmFtZSIsInByaW1UZW5hbnQiOnRydWUsInRva190eXBlIjoiQVQiLCJjYV9ndWlk
IjoiY2FjY3QtOGNjNmNmY2FlYzFjNDhjNGI0M2VjNWQ3M2RmYmE4MjciLCJhdWQiOlsidGVzdEFwcG
xpY2F0aW9uIiwiaHR0cHM6XC9cL2NvbnNvbGU6OTI0NSJdLCJ1c2VyX2lkIjoiY2RjMzE2ZDM0YWI5
NGZkOGEzZDZkODUyY2MwMDExMjYiLCJ0ZW5hbnRfaXNzIjoiaHR0cHM6XC9cL2lkY3MtMTdlY2EwMm
QzMWE2NDVlNzg3MTBhZDUyYWMwMWRkNDAuaWRlbnRpdHkuYzlkZXYyLm9jOXFhZGV2LmNvbTo0NDMi
LCJyZXNvdXJjZV9hcHBfaWQiOiI3YTVmMmI4OWI1ZDc0MWU4OGYyNDgxMjA3MDVlY2M1MSJ9.J6Mqg
piJJHEvHmQjGdhhabFgcctnhxCpIGe3GF6PJtI5fduhSUvBwodBv5wxlMMG-04ZlQQe85J-
THH8jpHpk42GiipwFvNMcERUGld7T99d5hazVJ20EOyguEDiUAqenXQ81H8eVwU9b8zzUcRJZFIw7D
WXgGoOVsQ9cDW7-KGQ3-
SatX0jQmEdbZn3n2LHJmv4k2uhiFZPol27iiYbXr6n5h7o3ZgZ8C9kPFkymNUAyVZp_7j8mqTQ6Qz0
7Gje1T-MOHIhKPA_ExceN95mbx4tvSPHGMz6DsEpo9r-
tcj9JDQcHFhOiT5gx-5hdNsrj9CZwLFgEf6m9n72CNFCzg
```

In this example, a credential called `dbnorth` is added with a token value as specified with the `TOKEN` parameter.

Refer to the Connect Command to connect to Oracle GoldenGate deployment using the credential name added to a secure store

ADD DISTPATH

Use `ADD DISTPATH` to create a distribution path. Distribution paths are created to link the Distribution Service of one deployment to the Receiver Service of another deployment, in order to send trails from the source to that target deployment.

Syntax

```
ADD DISTPATH path-name
         SOURCE source-uri
         TARGET target-uri|
         [TARGETTYPE ( MANAGER | COLLECTOR | RECVSRVR )] |
         AUTHENTICATION ( OAUTH
                        | (CERTIFICATE certificate-name)
```

```
| (USERIDALIAS alias [DOMAIN domain])
| [ENCRYPTIONPROFILE encryption-profile-name ]
```

path-name
The unique name of the distribution path you want to add.

source-uri
Specifies the source URI after the source keyword to indicate where the data is originated. The format of this URI contains the protocol (only supports trail), hostname, port number of the Distribution Service, and location of the source trail files.

target-uri
Specifies the target URI after the target keyword to indicate where the data will be sent to. The format of this URI is:
a supported protocol, `udt`, `ogg`, web socket `ws`, or secure web socket `wss`
a hostname,
the port number of the Receiver Service,
and the location of the target trail files.

TARGETTYPE
Specifies the target type in case the distribution path uses the legacy protocol. This argument is only valid if the target URI schema is `ogg://`.
Choose `MANAGER` if the target is a legacy deployment with a manager running.
Choose `COLLECTOR` if the target is a legacy deployment with a static collector running.
Choose `RECVSRVR` if the target is an Microservices Architecture deployment with a Receiver Service running.

AUTHENTICATION OAUTH
Use this option if you are using an external identity provider authorization profile. This will set up the flow from the Distribution Service to the Receiver Service. See Deploy-Delegate User Authentication and Authorization to an External ID Provider in the *Oracle GoldenGate Microservices Documentation*.

> **Note**
>
> If your deployment is enabled for IDCS, you can still choose to authenticate using other authentication options.

AUTHENTICATION CERTIFICATE *certificate-name*
Identifier of distribution path-specific client certificate uploaded and managed in Administration Service.

AUTHENTICATION USERIDALIAS
(Alternative to certificate authentication,) you can associate each distribution path with a target `DBLOGIN USERIDALIAS`.

ENCRYPTIONPROFILE
Specifies the name of the encryption profile for the distribution path.

Examples

```
ADD DISTPATH dpe  SOURCE trail://localhost:9002/services/v2/sources?trail=ea
                    TARGET wss://localhost:9003/services/v2/targets?trail=ea
```

The target trail must specify the directory that contains your trail files. The default `dirdat` directory is used in this example.

```
ADD DISTPATH dpe  SOURCE trail://localhost:9002/services/v2/sources?trail=ea
                    TARGET ogg://localhost:9003/services/v2/targets?
trail=dirdat/eb
                    TARGETTYPE MANAGER
```

A fully-qualified `DBLOGIN USERIDALIAS`.

```
ADD DISTPATH dpe SOURCE ... TARGET ws://recvsrvr-host:recvsrvr-port ...
AUTHENTICATION USERIDALIAS target-dblogin-useridalias DOMAIN target-domain
```

A `DBLOGIN USERIDALIAS` from a default domain.

```
ADD DISTPATH dpe SOURCE ... TARGET wss://recvsrvr-host:recvsrvr-port ...
```

An exmaple of this implementation is:

```
ADD DISTPATH dpe SOURCE ... TARGET wss://recvsrvr-host:recvsrvr-port ...
AUTHENTICATION USERIDALIAS ggeast DOMAIN OracleGoldenGate
```

You can alternatively choose to run the command with `base64-encoded user:password` for authenticating with the `USERIDALIAS` that is managed in the credential store.

```
ADD DISTPATH dpe SOURCE ... TARGET ws://webuser:webpass@recvsrvr-
host:recvsrvr-port...
```

In this example, `webuser` is the user and `webpass` is the password.

ADD ENCRYPTIONPROFILE

Use `ADD ENCRYPTIONPROFILE` to add an encryption profile for Oracle Key Vault using Oracle GoldenGate.

An encryption profile is used to retrieve a master key from an external Key Management Service (KMS). You can choose between Oracle Key Vault (OKV) and Oracle Cloud Infrastructure Keys Management Service (OCIKMS).

If there is no explicitly defined encryption profile, then a Local Wallet is used as the default profile.

To know more, see What is an Encryption Profile?.

Syntax

```
ADD ENCRYPTIONPROFILE encryption-profile-name
                    ( OKV
                      ( OKVPATH dir-path
                        [ KEYNAMEATTRIBUTE key-name-attribute ]
                        [ KEYVERSIONATTRIBUTE key-version-attribute ]
                        [ OKVVERSION okv-version ]
                        MASTERKEY
                        [ NAME ] masterkey-name
                        [ VERSION masterkey-version ]
                        [ TTL ttl ] )
                      | OCIKMS
                       ( ENDPOINT endpoint
                         TENANCY tenancy-ocid
                         USER user-ocid
                         APIKEY api-key-file
                         FINGERPRINT fingerprint
                         KEY key-ocid ) )
                      [ DEFAULT ]
```

OKV options:

OKVPATH
Specifies the directory where Oracle Key Vault client is installed.

KEYNAMEATTRIBUTE
Custom attribute used in Oracle Key Vault server to specify the masterkey name

KEYVERSIONATTRIBUTE
Custom attribute used in Oracle Key Vault server to specify the masterkey version.

OKVVERSION *okv-version*
Oracle Key Vault (OKV) version with possible values as 18.1 and 21.4.

MASTERKEY [NAME]
Name of the master key. This value must match the key name in the KMS parameter in Oracle GoldenGate and cannot be changed once replication has started.

MASTERKEY [VERSION]
Version of the master key. This must be a numeric value.

OCI KMS

Endpoint
Specify the Cryptographic endpoint of the OCI KMS Vault.

Tenancy
Specify the tenancy OCID.

User
Specify the OCID for the user.

APIKEY
Specify the path to the file containing the OCI API Signing Key. The key is read from this file and sent to Oracle GoldenGate. If you need to change the OCI API KEY at some point, you can use the ALTER ENCRYPTIONPROFILE command.

KEY
Specify the key OCID that belongs to a key stored in your OCI KMS Vault, this is the key used by Oracle GoldenGate for encryption (trail encryption).

Fingerprint
Hash (MD5) of the API signing key.

DEFAULT
Specifies the current encryption profile. If you set `DEFAULT YES` then the encryption profile is set to be the current encryption profile. If you set `DEFAULT NO` then the encryption profile is removed. If there is no explicitly defined current encryption profile (you set as `DEFAULT NO` to the previously current one) then the implicitly default profile is `LocalWallet`.

> **Note**
>
> Do not upload keys with duplicate values of `KeyName` and `KeyVersion`. At the time of startup, restart, or rollover, Oracle GoldenGate processes retrieve the highest `KeyVersion` value.

ADD EXTRACT

Valid for supported Db2, HP NonStop, MySQL, Oracle, PostgreSQL, SQL Server, Sybase ASE and Yugabyte databases.

Use `ADD EXTRACT` to create a source database capture process, known as an Extract. The Extract can be created as a change data Extract, which captures current transactions from the database log or through other means depending on the database vendor, or it can be created as an initial load Extract, which will capture the records that exist in the database tables.

It is recommended to create only one change data or one initial load Extract per source database, however in rare situations, it may improve capture throughput by using multiple Extracts per database.

Oracle GoldenGate can support a large number of concurrent Extract and Replicat processes per deployment, depending on the resources available with the operating system. However, it is recommended to monitor system resources as more Extract or Replicat processes get added, and to keep the total number of processes per deployment to 300 or less.

> **Note**
>
> - If you are creating an Extract with the same name as an old Extract, ensure that the old Extract has been completely removed and dropped. You can use the DELETE EXTRACT to remove the Extract and all it's components.
> - This command cannot exceed 500 bytes in size for all keywords and input, including any text that you enter for the `DESC` option.
> - For Db2 for i, this command establishes a global start point for all journals and is a required first step. After issuing the `ADD EXTRACT` command, you can then optionally position any given journal at a specific journal sequence number by using the `ALTER EXTRACT` command with an appropriate journal option.
> - For Oracle and PostgreSQL databases, establish a connection to the source database using DBLOGIN USERIDALIAS and then issue the `REGISTER EXTRACT` command before adding the Extract. For details, see the REGISTER EXTRACT command.
> - For SQL Server, establish a connection to the source database using `DBLOGIN USERIDALIAS`, prior to adding the Extract. This is necessary for the Administration Service to bind the credential information with the Extract process.

Syntax

```
ADD EXTRACT extract-name
|             { SOURCEISTABLE |
|               [ INTEGRATED ] TRANLOG
|                {, BEGIN     [ NOW | yyyy-mm-ddThh:mm.ssZ | LSN value | EOL |
EOF | EXTRBA archive-offset-number |  SCN  scn }
|               [, DESC        description    ]
|               [, CRITICAL    [ YES | NO ]     ]
|               [, ENCRYPTIONPROFILE     encryption-profile-name]
|               [, GTIDSET gtidset
|               [, PROFILE
|                 [ AUTOSTART   [ YES | NO ]
|                       [ DELAY     delay-number   ] ]
|                 [ AUTORESTART [ YES | NO ]
|                       [ RETRIES          retries-number ]
|                       [ WAITSECONDS      wait-number    ]
|                       [ RESETSECONDS     reset-number   ]
|                       [ DISABLEONFAILURE [ YES | NO ] ] ] ]
|               [ LOGNUM lognum]
|               [ LOGPOS logpos]
```

extract_name
The name of an Extract. It can contain up to eight characters, see Choosing Names for Processes and Files.

SOURCEISTABLE
Creates an Extract task that extracts entire records from the specified tables for an initial load using Oracle GoldenGate. When using `SOURCEISTABLE`, do not specify any service options. Task parameters must be specified in the parameter file, see Add Initial Load Extract Using the Admin Client in *Oracle GoldenGate Microservices Documentation*.

TRANLOG [*bsds_name*]
Use this option for all databases. TRANLOG requires the BEGIN option.
(Db2 on z/OS) You can use the *bsds_name* option for Db2 on a z/OS system to specify the Bootstrap Data Set file name of the transaction log, though it is not required and is not used. You do not need to change existing TRANLOG parameters.
(Oracle) Extract reads the Oracle redo logs directly. See INTEGRATED TRANLOG for an alternate configuration.

INTEGRATED TRANLOG
Valid for Oracle.
Adds Extract in integrated capture mode. In this mode, Extract integrates with the database logmining server, which passes logical change records (LCRs) directly to Extract. Before using INTEGRATED TRANLOG, use the REGISTER EXTRACT command.

BEGIN {NOW | *yyyy-mm-ddthh:mm.ssZ*}
Specifies a timestamp in the data source at which to begin processing.

> **Note**
>
> You must register the Extract before positioning the Extract to a specific timestamp.

- **NOW**

  NOW specifies the time at which the ADD EXTRACT command is issued and when the Extract is started, it will attempt to position to a transaction boundary at or after that specific time. Positioning an Extract by NOW is not as accurate as other methods, such as positioning by LSN, SCN, or LRI and requires that the database server and Oracle GoldenGate server be set to the same times and time zones.

  For Db2 LUW, only commit and end transaction records contain timestamps, so the Extract starting position can only be calculated relative to those timestamps. This is a limitation of the API that is used by Oracle GoldenGate. It must be noted that positioning by timestamp is not accurate and can also take a long time. It is recommended to use LRI or EOL options wherever possible.

- ***yyyy-mm-ddthh:mm.ssZ***

  A date and time (timestamp) in the given form. For example, 2017-07-14T14:54:45Z.

- ***yyyy-mm-ddT[hh:mi:[ss[.cccccc]]]Z***

  A date and time (timestamp) in the given form. For an Oracle Extract and an Extract for PostgreSQL, the timestamp value must be greater than the timestamp at which the Extract was registered with the database.

  Positioning by timestamp in PostgreSQL includes the following scenarios:

  – Scenario 1

    If track_commit_timestamp is off, the following output will be displayed when the Extract process starts irrespective of what positioning method is used:

    ```
    2020-04-29T02:15:54Z
    ```

  – Scenario 2

If the `track_commit_timestamp` is enabled before Extract is registered then the correct timestamp will be displayed once the records are pushed in the source database as mentioned in the following example:

```
2020-04-29 02:19:07 INFO OGG-01515 Positioning to begin time Apr
29,2020 2:18:38 AM.
```

- Scenario 3

  If `track_commit_timestamp` is enabled after the Extract is registered, then there may be chances that the older records are available in the log for which the commit timestamp is not built up with the associated `transaction ID`. In that case, if Extract does not get the timestamp then it will fallback using the default timestamp mentioned in scenario 1. The output will be similar to the following:

  ```
  020-04-29 01:55:07 INFO OGG-01517 Position of first record processed
  LSN: 0/221D028, Jan 1, 1970 12:00:00 PM.
  ```

- Past timestamp cannot be specified if the replication slot has moved away.

EXTRBA *archive-offset_number*
Valid for Db2 z/OS.
Specifies the relative byte address within a transaction log at which to begin capturing data. The required format is `0X`*nnn*, where *nnn* is a 1 to 20 digit hexadecimal number (the first character is the digit zero, and the second character can be upper or lower case letter `x`).

EOL
Valid for Db2 for i, Db2 LUW, MySQL, PostgreSQL, and SQL Server.
Configures processing to start at the end of the log files (or journals) where the next record will be written. Any active transactions will not be captured.
For Db2 LUW, it configures processing to start at the active `LRI` value in the log files. The active `LRI` is the position at the end of the log files that the next record will be written to. Any active transactions will not be captured.
For PostgreSQL, `DBLOGIN` is required for position by `EOL`.
For MySQL, it finds the position corresponding to the end of the file and starts reading transactions from there. The EOL position is not exact, if data is continuously written to the binary log.

EOF
Valid for MySQL, Db2 z/OS, Db2 LUW, SQL Server, PostgreSQL, Oracle GoldenGate for DAA.

LSN *value*
Valid for Db2 z/OS, PostgreSQL and SQL Server
Specifies the transaction `LSN` at which to start capturing data. An alias for this option is `EXTLSN`. Positioning to an LSN is precise.
For Db2 z/OS, LSN value can be either a series of hex digits or a series of decimal digits. If it is in hex format, it must be prefixed with either `0X` or `0x` followed by 1 to 20 hex digits: 0-9, a-g, or A-G. If it is in decimal format, there is no prefix and there are 1-25 decimal digits 0-9.
For PostgreSQL, `LSN` value can be `hi` or `lo`. Set the value as `hi` for the entry point of the log file. `Lo` is the offset in the log file. The `LSN` position should lie between the replication slot restart position and write ahead log current location. If the position specified itself exists between the mentioned range then Extract will throw an error.
For SQL Server, `LSN` specifies the transaction `LSN` at which to start capturing data. An alias for this option is `EXTLSN`.

The specified LSN should exist as a valid tran_begin_lsn found in the cdc.lsn_time_mapping system table, otherwise the Extract will attempt to position after the provided LSN value.
Valid LSN specifications for SQL Server consists of the following:

- Colon separated hex string (8:8:4) padded with leading zeroes and 0X prefix, as in 0X00000d7e:0000036b:0001
- Colon separated hex string with 0X prefix and without leading zeroes, as in 0Xd7e:36b:1

You can find the minimum LSN available by querying the following:

```
SELECT min([tran_begin_lsn]) FROM [cdc].[lsn_time_mapping] with (nolock)
where tran_id <> 0x00
```

The following examples show the use of LSN value for Db2 z/OS:
Example 1:

```
ADD EXTRACT extn TRANLOG, LSN 0xDEE40E4F27A3245400
```

Example 2:

```
ADD EXTRACT extn TRANLOG, LSN 22216433159121980904448
```

The following example shows the LSN value for SQL Server:

```
ADD EXTRACT extn TRANLOG, LSN 0X00000d7e:0000036b:0001
```

LRI *value*
Valid for Db2 LUW. Specifies a start position in the transaction logs when Extract starts.
You can use the LRI option for Db2 LUW systems to specify the LRI at which extract can start capturing records from the transaction log. You can use the Db2 utility db2logsForRfwd to obtain the LRI. This utility provides LRI ranges present in the Db2 logs.
Note that, although Extract might position to a given LRI, that LRI might not necessarily be the first one that Extract will process. There are numerous record types in the log files that Extract ignores, such as Db2 internal log records. Extract will report the actual starting LRI to the Extract report file.

LOGNUM *lognum*
Valid for MySQL.
This is the log file number. ADD EXTRACT will fail if the LOGNUM value contains zeroes preceding the value. For example, ADD EXTRACT ext1, TRANLOG, LOGNUM 000001, LOGPOS 0 will fail. Instead, set LOGNUM to 1 for this example to succeed.

LOGPOS *logpos*
This is an event offset value within the log file that identifies a specific transaction record. Event offset values are stored in the header section of a log record. To position at the beginning of a binlog file, set the LOGPOS as 0.

SEQNO *sequence_number*
Valid for Db2 for i. Starts capture at, or just after, a system sequence number, which is a decimal number up to 20 digits in length.

SCN *value*
Valid for Oracle.

Starts Extract at the transaction in the redo log that has the specified Oracle system change number (SCN). For Extract in integrated mode, the SCN value must be greater than the SCN at which the Extract was registered with the database. For more information, see REGISTER EXTRACT.

PARAMS *file_name*
Specifies the full path name of an Extract parameter file in a location other than the default of `dirprm` within the Oracle GoldenGate directory.

REPORT *file_name*
Specifies the full path name of an Extract report file in a location other than the default of `dirrpt` within the Oracle GoldenGate directory.

DESC '*description*'
Specifies a description of the group, such as `'Extracts account_tab on Serv1'`. Enclose the description within single quotes.

ENCRYPTIONPROFILE
Specifies the name of the Oracle GoldenGate encryption profile associated with the specific client.

GTIDSET *gtidset*
Valid for MySQL.
Specifies the initial positioning of Extract by using the position type as GTID set for GTID-based capture for MySQL. The supported MySQL sources for GTID set are MySQL Server 8.0 and higher. The maximum supported GTID set size is 64 KB.
Syntax:
`ADD EXTRACT extract_name, TRANLOG, GTIDSET gtidset`

CRITICAL
Indicates if the process is critical for the deployment.

PROFILE
Name of the auto start profile.

AUTOSTART
Specifies whether the managed process has to be started automatically when the Administration Service starts. The default value is `YES`.

DELAY
Time to wait in seconds before starting the process.

AUTORESTART
Controls how the process will be restarted if it terminates.

RETRIES
The maximum number of tries for restarting the task before canceling retry efforts. This is optional.

WAITSECONDS
The duration (in seconds) in which the retries are counted.

RESETSECONDS
Resets the duration in which the retries are counted.

DISABLEONFAILURE
If set to `TRUE`, then the task is disabled when the number of retries is exhausted.

SOCKSPROXY {*host_name* | *IP_address*}[:*port*] [PROXYCSALIAS *credential_store_alias* [PROXYCSDOMAIN *credential_store_domain*]
Use for an alias Extract. Specifies the DNS host name or IP address of the proxy server. You can use either one to define the host though you must use the IP address if your DNS server is unreachable. If you are using an IP address, use either an IPv6 or IPv4 mapped address, depending on the stack of the destination system. You must specify the `PROXYCSALIAS`. In addition, you can specify the port to use, and the credential store domain.

Examples

The following example creates an Extract group named `exte` that captures database changes from the transaction logs. Extraction starts with records generated at the time when the Extract group was created and started with `ADD EXTRACT`.

```
ADD EXTRACT extn, TRANLOG, BEGIN NOW
```

In the following example, an Extract group name `extw` is created to get the database changes from the transaction logs by positioning the Extract to a specific timestamp:

```
ADD EXTRACT extn, TRANLOG, BEGIN NOW


REGISTER EXTRACT extn DATABASE


ALTER EXTRACT extn BEGIN 2020-08-02T06:05:30.000Z
```

The following creates an integrated Extract group.

```
ADD EXTRACT extn, INTEGRATED TRANLOG, BEGIN NOW
```

The following creates an initial-load Extract named `extei`.

```
ADD EXTRACT extn, SOURCEISTABLE
```

The following examples include all the combinations of valid formats of GTID set.
Example 1:

```
ADD EXTRACT exte, TRANLOG, GTIDSET "E11FA47-71CA-11E1-9E33-C80AA9429562:4"
```

Example 2:

```
ADD EXTRACT exts, TRANLOG, GTIDSET "3E11FA47-71CA-11E1-9E33-C80AA9429562:1-10"
```

Example 3:

```
ADD EXTRACT extn, TRANLOG, GTIDSET "3E11FA47-71CA-11E1-9E33-
C80AA9429562:1-3:11:47-49"
```

Example 4:

```
ADD EXTRACT extw, TRANLOG, GTIDSET "2174B383-5441-11E8-B90A-
C80AA9429562:1-3,24DA167-0C0C-11E8-8442-00059A3C7B00:1-19"
```

> **Note**
>
> As shown in example 4, when GTID set contains multiple uuids, to form a REST API request, this GTID set is broken into multiple GTID sets using comma separator.

The following examples create and position Extract at a specific Oracle system change number (SCN) in the redo log.

```
ADD EXTRACT extn TRANLOG SCN 123456
```

```
ADD EXTRACT extn INTEGRATED TRANLOG SCN 123456
```

The following example creates an Extract on a Db2 LUW system.

```
ADD EXTRACT extn, TRANLOG LRI 8066.322711
```

The following example creates an Extract with the autostart option using Admin Client.

```
ADD EXTRACT extn, TRANLOG , BEGIN NOW , AUTOSTART yes
```

ADD EXTTRAIL

Use `ADD EXTTRAIL` to create a trail for online processing on the local system and:

- Associate it with an Extract group.
- Assign a maximum file size.
- When attempting to create multiple trail files using `ADD EXTTRAIL`, from a single Extract process, the operation will fail if the number of trail files exceed a certain limit. Specifically, when more than 12 trails are created, the following error may occur:

  ```
  2025-11-05T21:16:04Z  INFO    OGG-08100  Exceeded checkpoint record size
  (32,768/32,767) in
  checkpoint file /path/to/checkpoint/file.cpe.
  ```

  This error occurs due to the checkpoint file exceeding the size limit of 32,767 records. The checkpoint file contains CSN (Commit Sequence Number), XID (Transaction ID), and record numbers, which contribute to the file's size.

  To avoid this error it is recommended not to create multiple trail files from a single Extract, particularly for performance reasons. The preferred approach is to have a single Extract capture all required data and use the Distribution Service to distribute it to different destinations.

Another alternative option is to run multiple Extract processes if there is a need to reduce the trail file size for each individual destination.

See `EXTTRAIL` in the *Parameters and Functions Reference for Oracle GoldenGate*

Syntax

```
ADD EXTTRAIL trail_name, EXTRACT group_name
[, MEGABYTES n]
[SEQNO n]
```

trail_name
A two character trail name. It is recommended that both the characters be alphabets, but the first character must not be a number. Oracle GoldenGate appends this name with a nine-digit sequence number whenever a new file is created.
The trail name can also be prefixed with a directory. For example, a trail named `north/ea` would have trail files named `ea000000000`, `ea000000001` in the`$OGG_DATA_HOME/north` subdirectory.

group_name
The name of the Extract group to which the trail is bound. A trail can only be assigned to one Extract. Multiple Extracts cannot write to the same trail. However, one Extract can write to multiple distinct trails if needed, but this is not normally required.

MEGABYTES *n*
The maximum size, in megabytes, of each trail file in the sequence. The default is 500MB and the maximum value is 2000MB.

SEQNO*n*
Specifies that the first file in the trail will start with the specified trail sequence number. Do not include any zero padding. For example, to start at sequence 3 of a trail named `tr`, specify `SEQNO 3`. The actual file would be named `/ea000000003`. This option can be used during troubleshooting when a Replicat needs to be repositioned to a certain trail sequence number. It eliminates the need to alter the Replicat to read the required sequence number. The default value is 1.

Limitations

When creating an Extract trail with the same name as a prior Extract trail, make sure that all trails in that directory with the same `trail_name` have been deleted.

When capturing from a MySQL database enabled for group replication, only one trail is allowed for an Extract.

Examples

The following command creates a trail name `ea`:

```
ADD EXTTRAIL north/ea, EXTRACT exte, MEGABYTES 200
```

The output is displayed as follows:

```
2019-11-20T23:49:19Z  INFO    OGG-08100  EXTTRAIL added.
```

ADD HEARTBEATTABLE

Valid for all supported databases. See the Certification Matrix for details on supported databases.

This command requires a database login using `DBLOGIN`.

Use `ADD HEARTBEATTABLE` to create the objects necessary to use the automatic heartbeat functionality. This command performs the following tasks

- Creates a heartbeat seed table, heartbeat table, and heartbeat history table.
- Creates the `GG_LAG` and `GG_LAG_HISTORY` views.
- Creates the `GG_UPDATE_HB_TAB` and `GG_PURGE_HB_TAB` procedures that are called by the scheduler jobs.
- Creates the scheduler jobs that periodically update the heartbeat and seed table, and purge the history table. However, it does not create these jobs for Azure SQL Database, PostgreSQL, Teradata, and TimesTen.
- Populates the seed table.
- For Oracle multitenant databases:
  - The heartbeat objects and jobs are created in the user's schema that is connected to the database using the `DBLOGIN` command. Oracle GoldenGate Extract and Replicat look for the heartbeat objects in the `USERID` or `USERIDALIAS` schema. When making the connection using `DBLOGIN`, make sure that it is set to the appropriate `USERID` or `USERIDALIAS` schema that your Extract and Replicat processes will use.
  - Extract: Use the schema name in each PDB. Each PDB with which the Extract is registered should have its own heartbeat table.
  - Replicat: Each Replicat must have its own heartbeat table for its PDB.
  - For bidirectional, active/active replication, the heartbeat table should be in the same schema for the outgoing Extracts and incoming Replicats at each site. For example, see the following use case:

    Site A Site B

    `EAB ----------> RAB`

    `RBA ----------> EBA`

    In this example, `EAB` and `RBA` heartbeat tables must use the same schema. However, `EAB` and `RAB` can use different schemas.
- Oracle GoldenGate for Oracle heartbeat table administration has been simplified by eliminating the need for `GGSCHEMA` (or `HEARTBEATTABLE` parameter) except for limited circumstances. Heartbeat table administration operations are only done in the schema of the `DBLOGIN` user. Except for Oracle CDB root Extract, the Extract and Replicat processes look in the schema of the ER connected user for heartbeat tables. The following table shows the Extract and Replicat behavior for Oracle AI Database:

Extract/Replicat Processes	Behavior
Non-root Extract (non-CDB, and PDB)	If it is the first Extract user, then GGSCHEMA is used or in case of Autonomous AI Database, user must be ggadmin. If heartbeat is created after Extract starts, then look only in Extract user.
CDB Root Extract	GGSCHEMA is used.
Replicat	If it is the first Replicat user, then GGSCHEMA is used. For ADB, user must be ggadmin. If heartbeat is created after Replicat starts, then look only in the Replicat user.
DELETE EXTRACT/REPLICAT	If it is the first Extract or Replicat user, then GGSCHEMA is used. For ADB, user must be ggadmin.

This feature allows the usage of heartbeat tables in Oracle GoldenGate Hub deployments, where multiple databases are managed with differing Oracle GoldenGate administrator schemas.

> **Note**
>
> The heartbeat table objects should never be created in the root CDB of an Oracle Multitenant Database.

- For heterogeneous or non-Oracle databases, the heartbeat objects and jobs are created in the GGSCHEMA value listed in the GLOBALS file.

The default seed, heartbeat, and history table names are GG_HEARTBEAT_SEED, GG_HEARTBEAT, and GG_HEARTBEAT_HISTORY respectively.

In Microservices Architecture, the schema is configured using Replication Settings.

- The default names can be overridden by specifying HEARTBEATTABLE *hbschemaname.hbtablename* in the GLOBALS file.
- The tables, procedures, and jobs are created in the schema, *hbschemaname*.
- The seed and history table are created by appending a _SEED and _HISTORY to the table, *hbtablename*.

For Db2 LUW, you must set the DB2_ATS_ENABLE property with the db2set DB2_ATS_ENABLE=yes command.

For Db2 for i, to handle upgrade or misconfiguration of heartbeat table functionality, you can run the ADD HEARTBEATTABLE command again, which will repair the functionality of an exisiting heartbeattable setup without deleting the existing heartbeat data.

For Db2 for i, the licensed program, 5770SS1 Option 39 International Components for Unicode, must be installed.

For Db2 for i, the heartbeat table must be journaled to the same journal as the objects that are being replicated in the Extract using the said heartbeat table. If not, the Extract will abend indicating that more than one journal is available. In addition, any other Extract in the specific Oracle GoldenGate installation that is not reading the same journal cannot have the heartbeat table enabled for it.

For Amazon Aurora MySQL, the global variable `event_scheduler` must be enabled in the parameter group because Amazon RDS doesn't allow setting global variables. When the database is restarted, the `event_scheduler` returns to being disabled. To avoid this, you need to enable the `event_scheduler` in the `my.cnf/ini` file.

For PostgreSQL, a system job must manually be created to periodically call the heartbeat record update and history record purge function, `gg_hb_job_run`. For example, a cron job could be created that runs every minute. The function will check the actual heartbeat record update and purge frequency settings of the heartbeat configuration and only process operations within those boundaries:

```
PGPASSWORD="$passwd" psql -U gguser -d dbname -h dbhostname -p dbport# -c
"select <ggschema.gg_hb_job_run();" >/dev/null 2>&1
```

For Azure SQL Database, only `ADD HEARTBEATTABLE TARGETONLY` is supported and because there is no SQL Server Agent with Azure SQL Database, the purge routine cannot be created. So, the users must periodically run the `GG_PURGE_HB_TAB` stored procedure to remove old heartbeat records.

Syntax

```
ADD HEARTBEATTABLE
[, FREQUENCY number_in_seconds]
[, RETENTION_TIME number_in_days] |
[, PURGE_FREQUENCY number_in_days]
[, PARTITIONED]
[, TARGETONLY]
```

`FREQUENCY` *`number_in_seconds`*
Specifies how frequently the heartbeat records are generated. The default is 60 seconds. Consider the following limits:

- For Oracle AI Database, the minimum value is 0 and the maximum is 7999.
- For Db2 for i Series, the minimum value is 0 and the maximum is 7999.
- For Db2 LUW and Db2 z/OS, the minimum value is 60 and the maximum is 7999.
- The frequency for Db2 /zOS and Db2 LUW must be a multiple of 60 for values less than 3600 and multiples for 3600 for values greater or equal to 3600.
- For MySQL, the minimum value is 0 and the maximum is 7999.
- For SQL Server, the minimum value is 10 and the maximum is 7999.
- For PostgreSQL, the minimum value is 60 and the maximum is 7999.
- For Sybase, the minimum value is 1 minute (60 seconds) and maximum is 133 minutes (7999 seconds).
- Databases that support setting `FREQUENCY` to 0 will pause the heartbeat record scheduler.

`RETENTION_TIME` *`number_in_days`*
Specifies that heartbeat entries older than the retention time in the heartbeat history table are purged. The default is 30 days.
The minimum value for all databases is 1 and the maximum is 2147483646.

PURGE_FREQUENCY *number_in_days*
Specifies how often the purge scheduler is run to delete table entries that are older than the retention time from the heartbeat history table. The default is 1 day.
For Db2 LUW and Db2 z/OS, the minimum value is 1 and the maximum is 31.
For all other supported databases, the minimum value is 1 and the maximum value is 199.

PARTITIONED
Valid for Oracle.
Enables partitioning on the heartbeat history table. The column for the heartbeat time stamp received is used to partition the table with an interval of one day. By default the heartbeat history table is not partitioned.

TARGETONLY
Valid for Oracle AI Database, Db2 i Series, Db2 LUW, Db2 z/OS, MySQL, PostgreSQL, and SQL Server.

Does not enable supplemental logging on both the heartbeat seed and heartbeat tables and it does not create a scheduler job for updating the heartbeat table.

Examples

The following command creates default heartbeat tables, procedures and jobs.

```
ADD HEARTBEATTABLE
```

The following command creates the heartbeat tables, procedures and jobs with custom frequency, retention time, and purge frequency.

```
ADD HEARTBEATTABLE, FREQUENCY 120, RETENTION_TIME 10, PURGE_FREQUENCY 2
```

The following command creates the heartbeat tables, procedures and jobs with partitioning enabled in the heartbeat history table, and supplemental logging is not enabled in the heartbeat and heartbeat seed tables.

```
ADD HEARTBEATTABLE, partitioned, TARGETONLY
```

ADD MASTERKEY

Use the `ADD MASTERKEY` command to add a master key to a master-key wallet. The master key is used by Extract and Replicat to encrypt the encryption keys that secure data being sent across the network and in the trail files, so that those keys can be sent to, and used, by downstream processes. The master key omits the need to use wallet storage for the keys that actually encrypt the data.

The master-key wallet must be open to add a key.

The master key is generated as a random sequence of bits. The length is 256 bits by default. The key name is `OGG_DEFAULT_MASTERKEY`.

After adding a master key to a wallet that is not maintained centrally on shared storage, the updated wallet must be copied to all of the other systems in the Oracle GoldenGate configuration that use this wallet. Before doing so, Extract must be stopped and then all of the downstream Oracle GoldenGate processes must be allowed to finish processing their trails

and then be stopped. After the wallet is copied into place, the processes can be started again. For detailed instructions, see Encrypting Data with the Master Key and Wallet Method.

Syntax

```
ADD MASTERKEY
```

Example

```
ADD MASTERKEY
2019-11-21T19:37:23Z  ERROR   OGG-06137  Master key 'OGG_DEFAULT_MASTERKEY'
does not exist in Oracle Wallet.
2019-11-21T19:37:23Z  INFO    OGG-06142  Created version 1 of master key
'OGG_DEFAULT_MASTERKEY' in Oracle Wallet.
```

ADD PROFILE

This command is used to create a profile for managed Extract and Replicat processes from the Admin Client.

For Distribution and Receiver path processes, you need to use the `AUTOSTART` and `AUTORESTART` options available with the ALTER DISTPATH and ALTER RECVPATH commands.

Syntax

```
ADD PROFILE profile-name
             [ AUTOSTART        [ YES | NO ]
               [ DELAY            delay-number ]
             [ AUTORESTART      [ YES | NO ]
               [ RETRIES          retries-number]
               [ WAITSECONDS      wait-number]
               [ RESETSECONDS     reset-number]
               [ DISABLEONFAILURE [ YES | NO ] ]
```

`profile-name`
Name of the profile for the specific managed process.

`AUTOSTART`
Specifies whether the managed process has to be started automatically when the Administration Service starts. The default value is `YES`.

`DELAY`
Set the `delay-number` to configure the delay time to automatically start the managed process.

`AUTORESTART`
Specifies whether the managed process has to be restarted if it stops or abends. The default value is `YES`.

`RETRIES`
Specifies the number of retries for attempting to automatically restart the managed process.

`WAITSECONDS`
Specifies the time to wait before attempting another retry to restart.

RESETSECONDS
Used to reset the time for the retries.

DISABLEONFAILURE
Specifies if the managed process has to be disabled if it fails to restart. The default value is NO.

Example

```
ADD PROFILE Critical AUTOSTART AUTORESTART RETRIES 1 WAITSECONDS 0
RESETSECONDS 0 DISABLEONFAILURE NO

Command succeeded with no output.
```

ADD PROCEDURETRANDATA

Valid for Oracle.

Use ADD PROCEDURETRANDATA to add supplemental logging for Procedural Replication.

Syntax

```
ADD PROCEDURETRANDATA
```

ADD REPLICAT

Use ADD REPLICAT to create a Replicat group. This command creates an online process group that creates checkpoints so that processing continuity is maintained from run to run. It cannot exceed 500 bytes in size for all keywords and input, including any text that you enter for the DESC option.

Oracle GoldenGate supports up to 5,000 concurrent Extract and Replicat groups per instance of Oracle GoldenGate Service Manager. At the supported level, all groups can be controlled and viewed in full using commands such as the INFO and STATUS commands.

Oracle GoldenGate recommends keeping the number of Extract and Replicat groups (combined) at the default level of 300 or below in order to manage your environment effectively.

(Oracle) Unless the INTEGRATED option is used, this command creates a Replicat group in non-integrated mode.

Syntax

```
ADD REPLICAT group_name
[, PARALLEL [, INTEGRATED] | INTEGRATED | COORDINATED [MAXTHREADS number]]
{, EXTTRAIL trail_name}
[, BEGIN {NOW | yyyy-mm-dd[ hh:mi:[ss[.ccccccc]]]}]
[, EXTRBA rba]
{, CHECKPOINTTABLE owner.table}
[, DESC 'description']
[, ENCRYPTIONPROFILE encryption-profile-name ]
                [ CRITICAL        [ YES | NO ]  ]
                [ PROFILE          profile-name
```

```
[ AUTOSTART
                [ YES | NO ]
                [ DELAY delay-number ] ]
                [ AUTORESTART [ YES | NO ]
                [ RETRIES      retries-number ]
                [ WAITSECONDS  wait-number    ]
                [ RESETSECONDS reset-number   ]
                [ DISABLEONFAILURE [ YES | NO ]     ] ] ]
```

group_name
The name of the Replicat group. If you don't specify any option, then it creates a classic Replicat. The name of a coordinated and parallel Replicat group can contain a maximum of five characters. The name of a regular Replicat group can contain up to eight characters.

INTEGRATED
(Oracle) Creates the Replicat in integrated mode. Without this option, `ADD REPLICAT` creates the Replicat in non-integrated (classic) mode. This option works for parallel Replicat too. In this mode, the Replicat process leverages the apply processing functionality that is available within the Oracle AI Database. In this mode, Replicat operates as follows:

- Reads the Oracle GoldenGate trail.
- Performs data filtering, mapping, and conversion.
- Constructs logical change records (LCR) that represent source database DML or DDL transactions (in committed order).
- Attaches to a background process in the target database known as a database inbound server by means of a lightweight streaming interface.
- Transmits the LCRs to the inbound server, which applies the data to the target database.
- A Replicat in integrated mode (either normal integrated or parallel integrated) must be used if any of the following features are used:
  - Automatic conflict detection and resolution
  - Procedural replication
  - DML or DDL Handlers

`INTEGRATED` must be used for an online change-synchronization Replicat that reads from a local `EXTTRAIL`-specified trail.
Both integrated Replicat and parallel Replicat in integrated mode maintain the checkpoint table if it exists. Also see ADD CHECKPOINTTABLE.
`TRACETABLE` is not maintained by integrated Replicat or parallel Replicat in integrated mode.
When in integrated mode, Replicat does not support the following parameters:

- `BULKLOAD` (Do not use integrated Replicat as an initial-load Replicat.)
- `GENLOADFILES`
- `SHOWSYNTAX`
- `MAXTRANSOPS` (is ignored)

PARALLEL
Valid for all supported databases.
Adds the Replicat in parallel mode. In this mode, Replicat applies transactions in parallel to improve the performance. It takes into account dependencies between transactions. `PARALLEL INTEGRATED` Replicat is valid for Oracle only. It adds the parallel Replicat in integrated mode,

which like Integrated Replicat leverages the apply processing functionality that is available within the Oracle AI Database.

> **Note**
>
> When configuring parallel Replicat for Sybase, make sure that you upgrade to version 16.0 SP3 PL15 and higher to prevent the Administration Service from disconnecting intermittently.

COORDINATED [MAXTHREADS *number*]
Creates the Replicat in coordinated mode. A coordinated Replicat is multithreaded to enable parallel processing. This option adds the coordinator (identified by the group name) and the maximum number of processing threads that are specified by default or with MAXTHREADS. Dependencies are computed and coordinated by the coordinator, and the SQL processing is performed by the threads.
To create a COORDINATED Replicat, a checkpoint table is required.
COORDINATED must be used for an online change-synchronization Replicat that reads from a local EXTTRAIL-specified trail.

> **Note**
>
> The group name of a coordinated Replicat can contain only five characters.

MAXTHREADS *number*
Specifies the maximum number of processing threads that this Replicat group can spawn. These threads are all created on startup, but depending on what is specified in the MAP statements in the parameter file, some or all of these threads will process the workload at any given time. As a general rule, specify twice the number of threads that you specify in the MAP statements when you partition the workload. This allows you to add threads in the event that the workload increases, without having to drop and recreate the Replicat group, see TABLE | MAP for more information about how to partition the workload across threads. The default number of threads is 25 if MAXTHREADS is omitted. The maximum number of threads is 500.
MAXTHREADS has a relationship to the MAXGROUPS parameter. MAXGROUPS controls the maximum number of process groups (Extract and Replicat) allowed per instance of Oracle GoldenGate. Each Replicat thread is considered a Replicat group in the context of MAXGROUPS. Therefore, the number of Extract and Replicat groups in the Oracle GoldenGate instance, plus the value of MAXTHREADS, cannot exceed the value of MAXGROUPS, see MAXGROUPS

EXTTRAIL *trail_name*
Specifies the relative or fully qualified name of a trail that was created with the ADD EXTTRAIL command.

BEGIN {NOW | yyyy-mm-ddT*[hh:mm[:ss[.ccccc]]]*Z}
Defines an initial checkpoint in the trail.

NOW
Begins replicating changes from the time when the group is created.

***yyyy-mm-dd*T[*hh:mm[:ss[.ccccc]]]*Z**
Begins extracting changes from a specific time.

EXTRBA *extrba*
Valid for Db2 z/OS.
Specifies the relative byte address within the trail file that is specified by EXTSEQNO. Contact Oracle Support before using this option.

CHECKPOINTTABLE {*owner.table* | NODBCHECKPOINT}
Not valid for Oracle GoldenGate Applications Adapter or Oracle GoldenGate Big Data.
NODBCHECKPOINT is not valid for target Oracle AI Database.Oracle strongly recommends using a checkpoint table. Parallel and coordinated Replicat in integrated and non-integrated mode require a checkpoint table.
Specifies that this Replicat group will write checkpoints to the specified table in the database. Include the owner and table name, as in ggadmin.ggs_checkpoint. This argument overrides any default CHECKPOINTTABLE specification in the GLOBALS file. The table must first be added with the ADD CHECKPOINTTABLE command.
For non-Oracle target databases, when NODBCHECKPOINT is specified, then the Replicat group will not write checkpoints to a checkpoint table. This argument overrides any default CHECKPOINTTABLE specification in the GLOBALS file. This argument is required if you do not want to use a checkpoint table with the Replicat group that is being created.
For Oracle AI Database target, Oracle GoldenGate 26ai and higher will not support NODBCHECKPOINT parameter when adding a Replicat. After upgrading to Oracle GoldenGate 26ai, if the existing Replicat for Oracle AI Database was created with NODBCHECKPOINT and no default database checkpoint was specified in GLOBALS, then Replicat abends at startup. To avoid this, you must perform either of the following tasks:
Option 1: Oracle recommends that you specify a default database checkpoint table in GLOBALS
or
Option 2: Create a new Replicat with checkpoint table specified and restart from SCN that replaces upgraded Replicat.

DESC '*description*'
Specifies a description of the group, such as 'Loads account_tab on Serv2'. Enclose the description within quotes.

ENCRYPTIONPROFILE
Specifies the name of the encryption profile for the Replicat.

CRITICAL
Indicates if the process is critical for the deployment.

PROFILE
Name of the auto start profile.

AUTOSTART
Specifies whether the managed process has to be started automatically when the Administration Service starts. The default value is YES.

DELAY
Time to wait in seconds before starting the process.

AUTORESTART
Controls how the process will be restarted if it terminates.

RETRIES
The maximum number of tries for restarting the task before canceling retry efforts. This is optional.

WAITSECONDS
The duration (in seconds) in which the retries are counted.

RESETSECONDS
Resets the duration in which the retries are counted.

DISABLEONFAILURE
If set to TRUE, then the task is disabled when the number of retries is exhausted.

Examples

The following example adds an integrated Replicat named reps. The output is displayed using the INFO command.

```
ADD REPLICAT reps, INTEGRATED, EXTTRAIL ea, checkpointtable ggadmin.ckpt1
```

The output shows as:

```
2019-11-21T20:01:10Z  INFO    OGG-08100  REPLICAT (Integrated) added.
```

Here's an example of adding a parallel Replicat named repw.

```
ADD REPLICAT repw, INTEGRATED, PARALLEL, EXTTRAIL ea, checkpointtable
ggadmin.ggs_checkpoint
```

The output shows:

```
2019-11-21T20:07:26Z  INFO    OGG-08100  REPLICAT (Parallel) added.
```

ADD RMTTRAIL

Use ADD RMTTRAIL to create a trail for online processing by a Replicat on a remote, target system. The command:

- Assigns a maximum file size.
- Associates the trail with an Extract group.
- Uses remote trails in a Pump Extract.

Syntax

```
ADD RMTTRAIL trail_name, EXTRACT group_name
[, FORMAT RELEASE major.minor]
[, MEGABYTES n]
[, SEQNO n]
```

trail_name
For Microservices Architecture, a two character (alpha-numeric) maximum trail name, with no path listed, and the first character must not be a number. For example ea.
Oracle GoldenGate appends this name with a nine-digit sequence number whenever a new file is created. For example, a trail named ea would have trail files named ea000000000, ea000000001.

group_name
The name of the Extract group to which the trail is bound. A remote trail can only be assigned to one Extract. Multiple Extracts cannot write to the same trail. However, one Extract can write to multiple distinct remote trails.

MEGABYTES *n*
The maximum size, in megabytes, of a file in the trail. The default is 500, however, the value can be between 1 MB and 2000 MB (maximum).

SEQNO *n*
Specifies that the first file in the trail will start with the specified trail sequence number. Do not include any zero padding. For example, to start at sequence 3 of a trail named `ea`, specify `SEQNO 3`. The actual file would be named `ea000000003`. This option can be used during troubleshooting when Replicat needs to be repositioned to a certain trail sequence number. It eliminates the need to alter Replicat to read the required sequence number.

Example

Admin Client

```
ADD RMTTRAIL north/ea, EXTRACT exte, MEGABYTES 200
```

ADD RECVPATH

Use `ADD RECVPATH` to create a target-initiated distribution path in the Receiver Service.

Syntax

```
ADD RECVPATH path-name SOURCE source-uri TARGET target-uri
 [ENCRYPTIONPROFILE encryption-profile-name ]
```

path-name
The unique name of the distribution path you want to add.

source-uri
Specifies the source URI after the source keyword to indicate the location from where the has data originated. The format of this URI contains the protocol (only supports web socket `ws` protocol and secure web socket `wss` protocol), hostname, port number of the Distribution Service, and the location of the source trail files.

target-uri
Specifies the target URI after the target keyword to indicate the destination where the data will be sent to. The format of this URI contains the protocol (only supports trail), hostname, port number of Receiver Service, and location of the target trail files.

ENCRYPTIONPROFILE
This is the name (string) of the encryption profile for the Receiver path.

Examples

```
ADD RECVPATH rpe SOURCE ws://user:passwd@localhost:9002/services/v2/sources?
trail=ea TARGET
trail://localhost:9003/services/v2/targets?trail=da
```

ADD SCHEMATRANDATA

Valid for Oracle.

Use `ADD SCHEMATRANDATA` to enable schema-level supplemental logging for a table. `ADD SCHEMATRANDATA` acts on all of the current and future tables in a given schema to automatically log a superset of available keys that Oracle GoldenGate needs for row identification.

> **Note**
>
> Oracle GoldenGate 23.6 does not support enabling logical replication at the schema level for JSON Relational Duality Views (JSON DV) and JSON Collection Tables (JCTs).

To perform `ADD SCHEMATRANDATA` against a schema in the PDB of a multitenant database, you need to login to PDB to issue the command.

`ADD SCHEMATRANDATA` is valid for Extract and does the following:

- Enables Oracle supplemental logging for new tables created with a `CREATE TABLE`.
- Updates supplemental logging for tables affected by an `ALTER TABLE` to add or drop columns.
- Updates supplemental logging for tables that are renamed.
- Updates supplemental logging for tables for which unique or primary keys are added or dropped.
- Enables a table for auto-capture. The command add schema-level PK, UI, FK, ALLKEYS supplemental logging data.

By default, `ADD SCHEMATRANDATA` logs the key columns of a table in the following order of priority:

1. Primary key
2. In the absence of a primary key, all of the unique keys of the table, including those that are disabled, unusable or invisible. Unique keys that contain ADT member columns are also logged. Only unique keys on virtual columns (function-based indexes) are not logged.
3. If none of the preceding exists, all scalar columns of the table are logged. (System-generated row-OIDs are always logged.)

`ADD SCHEMATRANDATA` also supports the conditional or unconditional logging requirements for using integrated Replicat.

Use `ADD SCHEMATRANDATA` in the following cases:

- For all tables that are part of an Extract group that is to be configured for integrated capture. `ADD SCHEMATRANDATA` ensures that the correct key is logged by logging all of the keys.
- For all source tables that will be processed in an integrated Replicat group. Options are provided that enable the logging of the primary, unique, and foreign keys to support the computation of dependencies among relational tables being processed through different apply servers.
- When DDL replication is active and DML is concurrent with DDL that creates new tables or alters key columns. It best handles scenarios where DML can be applied to objects very shortly after DDL is issued on them. `ADD SCHEMATRANDATA` causes the appropriate key values to be logged in the redo log atomically with each DDL operation, thus ensuring metadata continuity for the DML when it is captured from the log, despite any lag in Extract processing.

Database-level Logging Requirements for Using `ADD SCHEMATRANDATA`

Oracle strongly encourages putting the source database into forced logging mode and enabling minimal supplemental logging at the database level when using Oracle GoldenGate. This adds row chaining information, if any exists, to the redo log for update operations.

Additional Considerations for Using `ADD SCHEMATRANDATA`

- Before using `ADD SCHEMATRANDATA`, issue the `DBLOGIN` command. The user who issues the command must be granted the Oracle Streams administrator privilege.

  ```
  EXEC DBMS_STREAMS_AUTH.GRANT_ADMIN_PRIVILEGE('GGADMIN')
  ```

- `ADD SCHEMATRANDATA` can be used instead of the `ADD TRANDATA` command when DDL replication is not enabled. Note, however, that if a table has no primary key but has multiple unique keys, `ADD SCHEMATRANDATA` causes the database to log all of the unique keys. In such cases, `ADD SCHEMATRANDATA` causes the database to log more redo data than does `ADD TRANDATA`. To avoid the extra logging, designate one of the unique keys as a primary key, if possible.
- For tables with a primary key, with a single unique key, or without a key, `ADD SCHEMATRANDATA` adds no additional logging overhead, as compared to `ADD TRANDATA`.
- If adding `SCHEMATRANDATA` on a schema in a PDB, you need to be logged into that PDB in `DBLOGIN`. For example, for `PDBEAST` and schema `HR`, use the following command:

  ```
  ADD SCHEMATRANDATA pdbeast.hr
  ```

- If you must log additional, non-key columns of a specific table (or tables) for use by Oracle GoldenGate, such as those needed for `FILTER` statements and `KEYCOLS` clauses in the `TABLE` and `MAP` parameters, issue an `ADD TRANDATA` command for those columns. That command has a `COLS` option to issue table-level supplemental logging for the columns, and it can be used in conjunction with `ADD SCHEMATRANDATA`.

Syntax

```
ADD SCHEMATRANDATA schema
{
[ALLOWNONVALIDATEDKEYS]
[NOSCHEDULINGCOLS | ALLCOLS]}
[NOVALIDATE]
```

```
[PARTIALJSON]
[PREPARECSN  {WAIT | LOCK | NOWAIT | NONE}]
```

schema
The schema for which you want the supplementary key information to be logged. Do not use a wildcard. To issue `ADD SCHEMATRANDATA` for schemas in more than one pluggable database of a multitenant container database, log in to each pluggable database separately with `DBLOGIN` and then issue `ADD SCHEMATRANDATA`. From the root conatiner, you may add schematrandata with the container prefix `ADD SCHEMATRANDATA [pdb_name].schema`
If you run the command from `cdb$root`, make sure that you also set the following user privilege on the database side:

```
ALTER USER userID SET CONTAINER_DATA = ALL  CONTAINER = CURRENT;
```

ALLOWNONVALIDATEDKEYS
It includes `NON VALIDATED` and `NOT VALID` primary keys in the supplemental logging. These keys override the normal key selection criteria that is used by Oracle GoldenGate. If the `GLOBALS` parameter `ALLOWNONVALIDATEDKEYS` is being used, `ADD SCHEMATRANDATA` runs with `ALLOWNONVALIDATEDKEYS` whether or not it is specified. By default `NON VALIDATED` and `NOT VALID` primary keys are not logged, see the `GLOBALS ALLOWNONVALIDATEDKEYS` parameter.

NOSCHEDULINGCOLS | ALLCOLS
These options control supplemental logging for an Oracle target database. You can use these options together though the latter option is used. For example, with the `ADD SCHEMATRANDATA oggadm_ext ALLCOL NOSCHEDULINGCOLS` command the `NOSCHEDULINGCOLS` option would be used.

NOSCHEDULINGCOLS
Disables the logging of scheduling columns. By default, `ADD SCHEMATRANDATA` enables the unconditional logging of the primary key and the conditional supplemental logging of all unique keys and foreign keys of all current and future tables in the given schema. Unconditional logging forces the primary key values to the log whether or not the key was changed in the current operation. Conditional logging logs all of the column values of a foreign or unique key if at least one of them was changed in the current operation. The integrated Replicat primary key, unique keys, and foreign keys must all be available to the inbound server to compute dependencies.
If you are enabling `auto_capture`, then do not use this option. This will allow tables in this schema to be auto captured unless the table is explicitly excluded/disabled for replication (such as through `TABLEEXCLUDE`, `DELETE TRANDATA`, or `ALTER TABLE DISABLE LOGICAL REPLICATION DDL`).

ALLCOLS
Enables the unconditional supplemental logging of all supported key and non-key columns for all current and future tables in the given schema. This option enables the logging of the keys required to compute dependencies, plus columns that are required for filtering, conflict resolution, or other purposes. Columns like LOB, LONG, and ADT are not included.

NOVALIDATE
Valid for all databases supported by `ADD SCHEMATRANDATA`.
Suppresses additional information about the table being handled being processed by `ADD SCHEMATRANDATA`. By default, this option is enabled. The additional information processing creates a lapse time on command response so this option can be used to increase response time.

PARTIALJSON
Valid for Oracle.
Fetches partial JSON updates at schema level. If enabled, the redo will include partial JSON records.
Also see, `TRANLOGOPTIONS FETCHPARTIALJSON` parameter.

PREPARECSN {WAIT | LOCK | NOWAIT | NONE}
Valid for Oracle for both DML and DDL.
Automatically prepares the tables at the source so the Oracle data pump Export dump file will includes Instantiation CSNs. Replicat uses the per table instantiation CSN set by the Oracle data pump (on import) to filter out trail records. On the target, the data pump import populates the system tables and views with instantiation SCNs using the `DBOPTIONS ENABLE_INSTANTIATION_FILTERING` parameter to enable table-level instantiation filtering.

WAIT
Wait for any in-flight transactions and prepare table instantiation.

LOCK
Put a lock on the table (to prepare for table instantiation).

NOWAIT
Default behavior, preparing for instantiation is done immediately.

NONE
No instantiation preparation occurs.

Example

The following enables supplemental logging for the schema `hr`.

```
ADD SCHEMATRANDATA hr
```

The following example logs all supported key and non-key columns for all current and future tables in the schema named `hr`.

```
ADD SCHEMATRANDATA hr ALLCOLS
```

The following example suppress additional table information processing.

```
ADD SCHEMATRANDATA hr NOVALIDATE
```

ADD TRANDATA

Use `ADD TRANDATA` to enable Oracle GoldenGate to acquire the transaction information that it needs from the transaction records.

Before using this command, use the `DBLOGIN` command to establish a database connection.

`ADD TRANDATA` is valid for the databases that are listed here:

- Db2 for i
- Db2 LUW
- Db2 z/OS

- Oracle
- PostgreSQL
- SQL Server
- Sybase

For other supported databases, this functionality may exist already or must be configured through the database interface.

See the following tables to know about the use of wildcards for different databases when using `ADD TRANDATA`:

Database	Wildcard Use
Db2 LUW, Db2 z/OS	Wildcard can be used with table names only. Not schema names.
Oracle	Wildcard can be used with schema and table names. Not PDB name.
PostgreSQL and SQL Server	Wildcard can be used with schema and table names

Db2 for i

Use `ADD TRANDATA` to start the journaling of data. The `ADD TRANDATA` command calls `STRJRNPF` and is the recommended method to start journaling for tables, because it ensures that the required journal image attribute of `Record Images (IMAGES): *BOTH` is set on the `STRJRNPF` command.

Db2 LUW

Use `ADD TRANDATA` to enable `DATA CAPTURE CHANGES` on specified tables. By default, `ADD TRANDATA` issues the following command to the database:

```
ALTER TABLE name DATA CAPTURE CHANGES INCLUDE LONGVAR COLUMNS;
```

You can exclude the `LONGVAR` clause by using `ADD TRANDATA` with the `EXCLUDELONG` option.

Db2 z/OS

Use `ADD TRANDATA` to enable `DATA CAPTURE CHANGES` on specified tables. By default, `ADD TRANDATA` issues the following command to the database:

```
ALTER TABLE name DATA CAPTURE CHANGES;
```

Oracle Database

Oracle GoldenGate 26ai for Oracle AI Database 26ai and higher support replication for JSON Relational Duality Views and JSON Collection Tables. The `ADD TRANDATA` command enables supplemental logging for JSON Relational Duality Views and JSON Collection Tables. However, `ADD TRANDATA` does not impact JSON Duality View base tables and will not add tranactional data for base tables. For details, see Prerequisites for Setting Up Replication for JSON DV and JSON Collection Tables in *Oracle GoldenGate Microservices Documentation*.

Note that using the command similar to the following, will only apply to tables and *NOT* to JSON Relational Duality Views and JSON Collection Tables:

```
ADD TRANDATA HR.*
```

To capture from JSON Duality Views and JSON Collection Tables, you need to run the `ADD TRANDATA` command for each JSON Duality View and JSON Collection Table that needs to be replicated. This ensures that you are aware of the JSON Duality Views and JSON Collection Tables that are being set up for replication to avoid system overload.

> **Note**
>
> Wildcards are not supported when enabling logical replication for JSON Duality Views using `ADD TRANDATA`. However, JSON Collection Tables support wildcards just like regular tables, therefore, the `TABLE/MAP` statement also allow use of wildcards.

From the 21c release onward, this command would also enable a table for auto capture for Oracle database.

By default, `ADD TRANDATA` for Oracle enables the unconditional logging of the primary key and the conditional supplemental logging of all unique key(s) and foreign key(s) of the specified table, see Ensuring Row Uniqueness in Source and Target Tables for more information about how Oracle GoldenGate handles supplemental logging for Oracle AI Database.

If possible, use the `ADD SCHEMATRANDATA` command rather than the `ADD TRANDATA` command. The `ADD SCHEMATRANDATA` command ensures replication continuity should DML occur on an object for which DDL has just been performed. You can exclude objects from the schema specification by using the exclusion parameters.

To use the Oracle GoldenGate DDL replication feature, you must use the `ADD SCHEMATRANDATA` command to log the required supplemental data.

When using `ADD SCHEMATRANDATA`, you can use `ADD TRANDATA` with the `COLS` option to log any non-key columns, such as those needed for `FILTER` statements and `KEYCOLS` clauses in the `TABLE` and `MAP` parameters.

> **Note**
>
> It is possible to use `ADD TRANDATA` for Oracle when DDL support is enabled, but only if you can stop DML on all tables before DDL is performed on them or, if that is not possible, you can guarantee that no users or applications will issue DDL that adds new tables whose names satisfy an object specification in a `TABLE` or `MAP` statement. There must be no possibility that users or applications will issue DDL that changes the key definitions of any tables that are already in the Oracle GoldenGate configuration.

Oracle recommends putting the source database into forced logging mode and enabling minimal supplemental logging at the database level when using Oracle GoldenGate. This adds row chaining information, if any exists, to the redo log for update operations

Take the following into account when using `ADD TRANDATA` for an Oracle AI Database:

- If any of the logging details change after Oracle GoldenGate starts extracting data, you must stop and then start the Extract process that is reading from the affected table before any data is changed.

- When creating a supplemental log group with ADD TRANDATA, Oracle GoldenGate appends the object ID to a prefix of GGS_, for example GGS_18342.

PostgreSQL

Using ADD TRANDATA command, the REPLICA IDENTITY setting of the table is altered. The REPLICA IDENTITY setting controls before images of columns of a table that should be logged to the transaction log for UPDATE and DELETE operations.

SQL Server

Use ADD TRANDATA to provide the extended logging information that Oracle GoldenGate needs to reconstruct SQL operations. The SQL Server transaction log does not provide enough information by default.

By enabling TRANDATA, Oracle GoldenGate enables the SQL Server Change Data Capture feature for the database and creates a Change Data Capture table for each instance enabled with TRANDATA.

Syntax

```
ADD TRANDATA container.owner.table]
[, FILEGROUP filegroup-name]
[, NOSCHEDULINGCOLS | ALLCOLS]
[, ALLOWNONVALIDATEKEYS]
[, PARTIALJSON]
[, PREPARECSN  {WAIT | LOCK | NOWAIT | NONE}]
```

[*container.*]*owner.table*
Valid for Db2 LUW, Db2 z/OS, Oracle, PostgreSQL, and SQL Server.
The two-part or three-part name specification. Use a two-part name of *owner.table* for all supported databases except an Oracle multitenant container database.
Use a three-part name of *container.owner.table* for an Oracle multitenant container database. A wildcard can be used for any component. Used with a wildcard, ADD TRANDATA filters out names that match the names of system objects. To use ADD TRANDATA for objects that are not system objects but have names that match those of system objects in a wildcard pattern, issue ADD TRANDATA for those objects without using a wildcard.
If you run the command from cdb$root, make sure that you also set the following user privilege on the database side:

```
ALTER USER userID SET CONTAINER_DATA = ALL  CONTAINER = CURRENT;
```

***schema.table* [JOURNAL *library/journal*] |**
***library/file* [JOURNAL *library/journal*]**
Valid for Db2 for i.
Specifies the SQL schema and name of a table or the native library and file name. If a default journal is set with the DEFAULTJOURNAL command, you can omit the JOURNAL option; otherwise it is required.

FILEGROUP *filegroup-name*
Valid for SQL Server.

(Optional) You can designate the filegroup in which the SQL Server Change Data Capture staging tables will be placed, by using the FILEGROUP option with an existing filegroup name.

```
ADD TRANDATA owner.table FILEGROUP cdctables
```

You can use the FILEGROUP option in the GLOBALS file also if you need to use the same FILEGROUP for each table when enabling TRANDATA.
The following example shows setting the FILEGROUP myFileGroup in GLOBALS file:
`FILEGROUP myFileGroup`
The output is:

```
Logging of supplemental log data is enabled for table dbo.test1 in filegroup
myFileGroup
```

In this case, ADD TRANDATA command uses the myFileGroup for all the tables.
If you also use the FILEGROUP parameter with ADD TRANDATA, the command overrides the filegroup name defined in the GLOBALS file.
For example, if you set FILEGROUP myFileGroup in the GLOBALS file and then execute the following ADD TRANDATA command:

```
ADD TRANDATA dbo.* FILEGROUP yourFileGroup
```

Then the output is:

```
Logging of supplemental log data is enabled for table dbo.test1 in filegroup
yourFileGroup
```

In this case, ADD TRANDATA uses the yourFileGroup instead of myFileGroup for all the tables.
If you don't specify the FILEGROUP either in GLOBALS or with ADD TRANDATA, then the command line considers the default FILEGROUP of the database while adding TRANDATA of the table. For example, if you run ADD TRANDATA dbo.*, the output is:

```
Logging of supplemental log data is enabled for table dbo.test1 in filegroup
PRIMARY
```

In this case, the default FILEGROUP is Primary. If you run the INFO TRANDATA command, the FILEGROUP name shows as PRIMARY.

NOSCHEDULINGCOLS | ALLCOLS
Valid for Oracle and PostgreSQL.
From Oracle GoldenGate 21c onward, NOSCHEDULINGCOLS is deprecated for Oracle database 21c and higher if the running database supports auto capture capabilities.
These options satisfy the logging requirements of an integrated Replicat that will be processing the tables that you are specifying with ADD TRANDATA.

NOSCHEDULINGCOLS
Valid for Oracle only.
Disables the logging of scheduling columns. By default, ADD TRANDATA enables the unconditional logging of the primary key and the conditional supplemental logging of all

unique keys and foreign keys of the specified table. Unconditional logging forces the primary key values to the log whether or not the key was changed in the current operation. Conditional logging logs all of the column values of a foreign or unique key if at least one of them was changed in the current operation. The primary key, unique keys, and foreign keys must all be available to the inbound server to compute dependencies.

ALLCOLS
Enables the unconditional supplemental logging of all of the key and non-key columns of the table. This option enables the logging of the keys required to compute dependencies, plus all other columns for use in filtering, conflict resolution, or other purposes.

For PostgreSQL, ALLCOLS sets REPLICA IDENTITY for the table to FULL. ALLCOLS is specified as part of the ADD TRANDATA command, to enable logging of all the columns for UPDATE and DELETE operations, even if those columns have not been modified. For tables without a Primary Key or Unique Index, the ALLCOLS option is redundant. Here's the syntax:

```
ADD TRANDATA table_name ALLCOLS
```

COLS (*columns*)
Valid for Oracle.
Use the COLS option to log specific non-key columns. Can be used to log columns specified in a KEYCOLS clause and to log columns that will be needed for filtering or manipulation purposes, which might be more efficient than fetching those values with a FETCHCOLS clause in a TABLE statement. Separate multiple columns with commas, for example NAME, ID, DOB.

INCLUDELONG | EXCLUDELONG
Valid for Db2 LUW.
Controls whether or not the ALTER TABLE issued by ADD TRANDATA includes the INCLUDE LONGVAR COLUMNS attribute. INCLUDELONG is the default. When ADD TRANDATA is issued with this option, Oracle GoldenGate issues the following statement:

```
ALTER TABLE name DATA CAPTURE CHANGES INCLUDE LONGVAR COLUMNS;
```

When EXCLUDELONG is used, the following is the command:

```
ALTER TABLE name DATA CAPTURE CHANGES;
```

When EXCLUDELONG is used, Oracle GoldenGate does not support functionality that requires before images of tables that include LONGVAR columns. For example, the GETUPDATEBEFORES parameter. To support this functionality, changes to LONGVAR columns in the transaction logs must include both the before and after images of the column value.

NOKEY
Valid for Db2 for i, Db2 LUW, Db2 z/OS, Oracle.
From Oracle GoldenGate 21c onward, NOKEY is deprecated for Oracle Database 21c and higher if the running database supports auto capture capabilities.
Suppresses the supplemental logging of primary key columns. If using NOKEY, use the COLS option to log alternate columns that can serve as keys, and designate those columns as substitute keys by using the KEYCOLS option of the TABLE or MAP parameter.

NOVALIDATE
Valid for Db2 for i, Db2 LUW, Db2 z/OS, Oracle.

Suppresses additional information about the table being handled being processed by `ADD TRANDATA`. By default, this option is enabled. The additional information processing creates a lapse time on command response so this option can be used to increase response time.

`ALLOWNONVALIDATEDKEYS`
Valid for Db2 for i, Db2 LUW, Db2 z/OS, Oracle.
It includes `NON VALIDATED` and `NOT VALID` primary keys in the supplemental logging. These keys override the normal key selection criteria that is used by Oracle GoldenGate. If the `GLOBALS` parameter `ALLOWNONVALIDATEDKEYS` is being used, `ADD SCHEMATRANDATA` runs with `ALLOWNONVALIDATEDKEYS` whether or not it is specified. By default, `NON VALIDATED` and `NOT VALID` primary keys are not logged.

`PARTIALJSON`
Valid for Oracle.
Fetches partial JSON updates at the table level. If enabled, the redo will include partial JSON records.
Also see, `TRANLOGOPTIONS FETCHPARTIALJSON` parameter.

`PREPARECSN {WAIT | LOCK | NOWAIT | NONE}`
Valid for Oracle for both DML and DDL. Automatically prepares the tables at the source so the Oracle data pump Export dump file will includes Instantiation CSNs. Replicat uses the per table instantiation CSN set by the Oracle data pump (on import) to filter out trail records. On the target, the data pump import populates the system tables and views with instantiation SCNs using the `DBOPTIONS ENABLE_INSTANTIATION_FILTERING` parameter to enable table-level instantiation filtering.

`WAIT`
Wait for any in-flight transactions and prepare table instantiation.

`LOCK`
Put a lock on the table (to prepare for table instantiation).

`NOWAIT`
Default behavior, preparing for instantiation is done immediately.

`NONE`
No instantiation preparation occurs.

Examples

The following example causes one of the following: the primary key to be logged for an Oracle table; supplemental data to be logged for a SQL Server or DB2 table. This would also enable the table for auto capture.

```
ADD TRANDATA hr.employees
```

The following example enables the unconditional supplemental logging of all of the key and non-key columns for the table named `acct`. This would also enable the table for auto capture.

```
ADD TRANDATA hr.employees ALLCOLS
```

The following Oracle AI Database example causes the primary key to be logged plus the non-key columns name and address. This would also enable the table for auto capture.

```
ADD TRANDATA hr.employees, COLS (name, address)
```

The following Oracle AI Database example prevents the primary key from being logged, but logs the non-key columns `name` and `pid` instead. This would also enable the table for auto capture.

```
ADD TRANDATA hr.employees, NOKEY, COLS (name, pid)
```

The following example adds logging although it does not prepare the table for instantiation. This would also enable the table for auto capture.

```
ADD TRANDATA hr.employees PREPARECSN NONE
```

The following example suppresses additional table information processing. This would also enable the table for auto capture.

```
ADD TRANDATA hr.employees.*name NOVALIDATE
```

ALTER AUTHORIZATIONPROFILE

Alters or enables an authorization profile in a specified deployment. If a non-`localCredentialStore` profile is enabled, a communication and configuration test is triggered. If the test passes, the profile is set, otherwise no change occurs. You need to restart your Microservices deployment, before this profile gets loaded. Service Manager automatically loads the profile.

The `localCredentialStore` profile can be set, but its content cannot be changed.

Syntax

```
ALTER AUTHORIZATIONPROFILE profile-name
      DEPLOYMENT deployment-name
        ( [ ENABLE ]
          [ ID client-id [ SECRET client-secret ] ]
          [ DISCOVERYURI tenant-discovery-uri ]
          [ GROUPS
              ( SECURITY security-group
                | [ ADMINISTRATOR administrator-group ]
                | [ OPERATOR operator-group ]
                | [ USER user-group ]
              ) ]
          [ TTLSECONDS ttl-number ]
          [ DESCRIPTION description ]
        )
```

ENABLE
Use this option to enable the profile.

The other options used in this command are the same as the ADD AUTHORIZATIONPROFILE command.

Example

The following example shows how to enable a profile using this command:

```
ALTER AUTHORIZATIONPROFILE apn DEPLOYMENT ServiceManager ENABLE
```

The following example shows how to clean the administrator group name.

```
ALTER AUTHORIZATIONPROFILE apn DEPLOYMENT ServiceManager GROUPS ADMINISTRATOR
''
```

The following example shows how to update the client ID and secret.

```
ALTER AUTHORIZATIONPROFILE apn DEPLOYMENT ServiceManager
CLIENT ID 1234567890abcdefghijklmnopqrstuv SECRET 12345678-90ab-cdef-ghij-
klmnopqrstuv
```

ALTER CREDENTIALSTORE

Use this command to create credential store aliases which contains database credentials that includes username and password with the connection details. The credential store aliases can then be used to establish database connections within deployment.

See Add and Alter Database Credentials to configure the database connections from the MA web interface.

The use of a credential store is not supported for the NonStop platforms.

Syntax

```
ALTER CREDENTIALSTORE
{ADD USER {userid|userid@dbhost:dbport/dbname[;connection_options]|
userid@odbc-dsn}
[DELETE USER alias]
[REPLACE USER alias]
[ALIAS alias]
[Domain domain]
[PASSWORD password|NOPASSWORD]
```

ADD USER {userid|userid@dbhost:dbport/dbname[;connection_options]|userid@odbc-dsn}
userid
Adds the specified user to the credential store alias. If the `ALIAS` option is not used, the alias name defaults to that of the *`userid`*. If the `DOMAIN` option is not supplied, the domain defaults to the OracleGoldenGate domain. For multitenant Oracle databases with different users for the CDB and the PDB, you need to specify *`@tns_service_name`* when adding a database user to the credential store.

userid@dbhost:dbport/dbname[;connection_options]
For databases that support connection strings such as Oracle database, MySQL, SQL Server, PostgreSQL, and Db2, adds the specified user and connection information to a credential store alias. Connection string information should include the database server name, database port, and database name. Following are the additional connection options (per database) that are available to add to the connection string:
The `ALIAS` option is required when creating an alias with database connection details
The required connection format is:
`userid@databasehost:databaseport/databasename;option1;option2;option3`
For example:

```
ALTER CREDENTIALSTORE ADD USER ggadmin@server_west:1434/sourcedb
PASSWORD ***** ALIAS SourceMSSQLConnection
```

userid @ odbc-dsn
For databases that support DSN connections over ODBC, including Db2 for i Series, Db2 LUW, Db2 z/OS, SQL Server, and PostgreSQL, adds the DSN connection information to the credential store. The DSN connection details must exist on the system, such as in an `odbc.ini` file for Linux or as a System DSN created under Windows using the ODBC data sources 64-bit client.
The `ALIAS` option is required when creating an alias with a DSN.
The required connection format is:
`userid@odbc-dsn`
For example:

```
ALTER CREDENTIALSTORE ADD USER ggadmin@sourcedsn
PASSWORD ***** ALIAS SourceMSSQLConnection
```

DELETE USER *alias*
Removes the credential of the specified alias. If the `ALIAS` option is not supplied, the alias name defaults to that of the `userid`. If the `DOMAIN` option is not supplied, the credential store defaults to the `OracleGoldenGate` domain.
For example:

```
ALTER CREDENTIALSTORE DELETE USER src_alias
```

REPLACE USER *userid*
Replaces the user of a given alias or can be used to change the password of an existing user. If the `ALIAS` option is not supplied, then the alias name defaults to that of the `userid`. If the `DOMAIN` option is not supplied, then the credential store defaults to the OracleGoldenGate domain.
Unless the `PASSWORD` option is used, the command prompts to enter a password for the specified user. When changing the password for a user that includes database connection information, you must include the full value of the current `userid`, otherwise this command will replace the user with the new value supplied.
The following example shows how to change the password of an existing user `ggadmin@sourcedsn`, alias `SourceMSSQLConnection`.

```
ALTER CREDENTIALSTORE REPLACE USER ggadmin@sourcedsn PASSWORD newpassword
ALIAS SourceMSSQLConnection
```

The example shows how to change the user ggadmin of the pdbeast alias to a new user, which then prompts for the password.

```
ALTER CREDENTIALSTORE REPLACE USER ggadmin ALIAS pdbeast DOMAIN Production
```

ALIAS *alias*
Specifies an alias for the user name. Use this option if you do not want the user name to be in a parameter file or command. If ALIAS is not provided, the alias defaults to the USER name, which then must be used in parameter files and commands where a login is required. You can create multiple entries for a user, each with a different alias, by using the ADD USER option with ALIAS.

DOMAIN domain
Saves the credential alias under the supplied domain name.
The default domain is OracleGoldenGate. By choosing unique domain names, the same alias can be used by multiple Oracle GoldenGate installations that use the same credential store. For example, the administrators of system 1 might not want system 2 to have access to the same credentials that are used on system 1. Those credentials can be stored as ALIAS pdbeast, for example, under DOMAIN system1, while a different set of credentials can be stored for ALIAS pdbeast under DOMAIN system2.

NOPASSWORD | PASSWORD *password*
Specify the user's password using the PASSWORD option. The password is echoed (not obfuscated) when this option is used. If this option is omitted, the command prompts for the password, which is obfuscated as it is typed (recommended as more secure).
Oracle GoldenGate 26ai supports a maximum password length of 1024 bytes.
The NOPASSWORD option is the alternative to the PASSWORD option when using external authentication because password is not required for external authentication such as using Kerberos authentication or IDCS. After the NOPASSWORD option is set, the DBLOGIN command can be used to access the database without a password.
Also see USERIDALIAS parameter in the *Parameters and Functions Reference for Oracle GoldenGate*.

Connection Options
These are optional connection string settings that can be added to a credential alias, including the SSL attributes. They must be specified as semicolon separated list of attributes, for example, option1=value1;option2=value2;option3=value3, where the options are the database specific properties along with their allowed values.
Following are a list of additional connection string options that can be added to a credential alias, including the SSL attributes.

> **Note**
>
> Oracle GoldenGate Microservices Architecture supports SSL connections by configuring certain SSL parameters as environment variables. However, this configuration limits the Extract and Replicat process to connect using only the defined SSL environment variables for a deployment and no other SSL or Non-SSL connections are possible. However, a credential store alias also provides the flexibility of configuring the credentials with different SSL or non SSL parameters for different database connections within the same deployment.

sslCA
For MySQL and PostgreSQL only. The certificate used to verify the authenticity of the SSL certificates in the PEM format.

sslCert
For MySQL and PostgreSQL. The client's SSL certificate file in the PEM format.

sslCrl
For MySQL. The name of the file containing certification revocation lists in the PEM format.

sslKey
For MySQL and PostgreSQL. The private key file of client in the PEM format.

sslMode
For MySQL and PostgreSQL. The sslMode parameter values are specified in uppercase only. The sslMode options for MySQL are `VERIFY_IDENTITY, VERIFY_CA, REQUIRED, PREFERRED`. The sslMode options for PostgreSQL are `VERIFY-FULL, VERIFY-CA, REQUIRE, PREFER`. For `sslMode=VERIFY_IDENTITY`, it is mandatory to provide sslPath, sslCa, sslCert and sslKey. For `sslMode=VERIFY_CA`, it is mandatory to provide sslPath and sslCa.

sslPath
For MySQL only. The path of the directory that contains trusted ssl Certificate Authority (CA) certificate files in PEM format.

Examples

Oracle

This example adds a user named `ggadmin` but omits the `PASSWORD` specification, so the command prompts for the password of the pluggable database, `pdbeast`.

```
ALTER CREDENTIALSTORE ADD USER ggadmin
Password: ********
```

This example adds the user `ggadmin` with this password `tiger` and specifies the alias as `pdbeast`.

```
ALTER CREDENTIALSTORE ADD USER ggadmin PASSWORD tiger ALIAS pdbeast
```

This example adds the user `ggadmin` under the domain of `OracleGoldenGate`.

```
ALTER CREDENTIALSTORE ADD USER ggadmin@dbnorth ALIAS pdbeast DOMAIN
OracleGoldenGate
Password: ********
```

This example adds the user `ggadmin` using the Easy Connect Naming Method.

```
ALTER CREDENTIALSTORE ADD USER ggadmin@dbnorth:1521/dbservice_north
ALIAS pdbnorth
```

This example issues two `ALTER CREDENTIALSTORE` commands, each of which adds a `ggadmin` entry, but with a different alias.

```
ALTER CREDENTIALSTORE ADD USER ggadmin@dbeast ALIAS pdbeast
Password: ********
ALTER CREDENTIALSTORE ADD USER ggadmin@dbwest ALIAS pdbwest
Password: ********
```

The following shows how the `DELETE USER` option works with and without the `ALIAS` option. The following command deletes the `user1` entry for which the `ALIAS` is the same as the user name.

```
ALTER CREDENTIALSTORE DELETE USER ggadmin
Alias: pdbeast
Userid: ggadmin
```

The following command deletes the entry for user `ggadmin` that is associated with the alias `pdbeast`.

```
ALTER CREDENTIALSTORE DELETE USER ggadmin ALIAS pdbeast
Alias: pdbeast
Userid: ggadmin
```

This example uses a SQL*Net connect string as the user value. In this case, the `PASSWORD` option is omitted. The person issuing the command is prompted for the password, which is hidden.

```
ALTER CREDENTIALSTORE ADD USER ggadmin@pdbeast ALIAS pdbeast
```

This example creates a domain name Oracle GoldenGate with user ID `ggadmin` and alias as `pdbeast` in the Admin Client.

```
ALTER CREDENTIALSTORE ADD USER ggadmin ALIAS pdbeast
Password:
```

Following example connects using a connection qualifier if using a `BEQ-Bequeath Protocol` adapter. For more information on establishing a secure connection using a `BEQ-Bequeath Protocol` adapter, see Configure Secure Database Connection

```
ALTER CREDENTIALSTORE ADD USER ggadmin@inst1_beq
```

The following example (Admin Client) adds a user named `ggadmin` but with external authentication and therefore uses the `NOPASSWORD` option.

```
ALTER CREDENTIALSTORE ADD USER ggadmin@dbeast nopassword alias pdbeast
```

Running the INFO CREDENTIALSTORE command, you can check the add user to the credential store:

```
INFO CREDENTIALSTORE
Default domain: OracleGoldenGate
  Alias: pdbeast
  Userid: @ggadmin
```

After you update the credential store to use the NOPASSWORD option, you can use the DBLOGIN command with Kerberos authentication for your database.

```
DBLOGIN USERIDALIAS pdbeast
```

MySQL
Following are some examples to create credential in Admin Client for MySQL.

This example adds database credentials with no SSL attribute in the connection string.

```
ALTER CREDENTIALSTORE ADD USER ggadmin@mysqlsrvr.myorg.com/my_db
PASSWORD:******  ALIAS pdbeast
```

This example adds database connection details in the credential store with SSL attributes added in the connection string.

```
ALTER CREDENTIALSTORE ADD USER ggadmin1@192.168.254.2:3306/atssrc;
sslMode=VERIFY_IDENTITY;sslPath=/var/lib/mysql/data;
sslCert=client-cert.pem;sslCa=ca.pem;sslKey=client-key.pem
PASSWORD ****** ALIAS pdbeast
```

SQL Server
The following example adds database connection details in the credential store using the DSN.

```
ALTER CREDENTIALSTORE ADD USER ggadmin@mydsn PASSWORD *****
ALIAS pdbeast
```

The following example adds database connection details in the credential store using the DSNless based connection.

```
ALTER CREDENTIALSTORE ADD USER ggadmin@100.70.96.45:10043/qadb;Encrypt=NO;
TrustServerCertificate=YES PASSWORD***** ALIAS pdbwest
```

PostgreSQL
The following example adds database connection details in the credential store with SSL attributes added in the connection string.

```
ALTER CREDENTIALSTORE ADD USER ggadmin@abc.com:7432/mydb;sslmode=disable;
sslCa=/u01/ogg/certs/root.pem  PASSWORD ***** ALIAS pdbeast
```

The following example adds database connection details in the credential store using the DSN.

```
ALTER CREDENTIALSTORE ADD USER ggadmin@mydsn PASSWORD *****
ALIAS pdbeast
```

The following example adds database connection details in the credential store using the DSNless based connection.

```
ALTER CREDENTIALSTORE ADD USER ggadmin@100.70.96.45:10043/qadb PASSWORD*****
ALIAS pdbwest
```

Sybase
Following are some examples to create credential in Admin Client for Sybase.

The following example displays a host and port based connection with no additional attributes to the Sybase database.

```
ALTER CREDENTIALSTORE ADD USER ggadmin@sybdb.myorg.com:5055/qadb
PASSWORD****
ALIAS pdbeast
```

The following example displays a host and port based connection to the Sybase database with additional attributes and SSL enabled.

```
ALTER CREDENTIALSTORE ADD USER ggadmin@sybdb.myorg.com:5055/qadb;OGG_USE_SSL;
CS_OPT_CHARSET=UTF-8;CS_LOGIN_TIMEOUT=10 PASSWORD**** ALIAS pdbeast
```

DB2 for i and DB2 for z/OS
The following example adds database connection details in the credential store using the DSN.

```
ALTER CREDENTIALSTORE ADD USER ggadmin@mydsn PASSWORD *****
ALIAS pdbeast
```

The following example adds database connection details in the credential store using the DSNless based connection.

```
ALTER CREDENTIALSTORE ADD USER ggadmin@100.70.96.45:10043/qadb PASSWORD*****
ALIAS pdbwest
```

ALTER DISTPATH

Use `ALTER DISTPATH` to change the attributes of a distribution path.

Syntax

```
ALTER DISTPATH path-name
        ( BEGIN ( NOW
                  | SEQNO trail-sequence-number RBA relative-byte-address
                  | begin-datetime ),
```

```
        | SOURCE source-uri),
        | TARGET target-uri),
                  | FORMAT ( CANONICAL | TEXT | SQL | XML
                  | SIZEMB megabytes-number
                  | SEQLEN sequence-length
                  | PROXY URI proxy-uri
                         TYPE ( SOCKS | HTTP
                         [ CSALIAS credential-store-alias)
                           [ CSDOMAIN credential-store-
domain ] ]
                  | COMPRESSION ( ON | OFF | THRESHOLD compression-threshold )
  |               | RULE    ( PASSTHRU
  |                   | FILTER [ CHUNKIDS    chunk-ids ]
  |                            [ OBJECTNAMES object-names-wildcard ]
  |                            [ OBJECTTYPES [ DML ] [ DDL ] [ PROCEDURE ] ]
  |                            [ TAGS        binary-tags ]
  |                            [ PROCEDUREFEATURENAMES feature-names-
wildcard ]
  |                            [ COLUMNVALUES column-values ]
  |                            [ PARTITIONNAMES partition-names-wildcard ] )
  |                   [ RELATION    ( AND  | OR  ) ]
  |                   [ ACTION      ( INCLUDE | EXCLUDE ) ]
  |         | OPTIONS [ AUTORESTART RETRIES retries [ DELAY delay ] ]
  |                   [ CRITICAL ( YES | NO ) ]
  |                   [ EOFDELAY    eofdelay         ]
  |         | AUTHENTICATION ( OAUTH
                                  | ( CERTIFICATE certificate-name )
                                  | ( USERIDALIAS alias [DOMAIN domain
                                  | ( ENCRYPTIONPROFILE encryption-profile-
name] ) ) )
```

path-name
The name of the distribution path you want to change.

BEGIN {NOW | SEQNO *sequence-number* RBA *relative-byte-address* | *begin-datetime*}
Specifies a timestamp in the data source at which to begin processing.

NOW
Specifies the time at which the `ADD EXTRACT` command is issued.

sequence-number relative-byte-address
The sequence number of the trail file and RBA within that file at which to begin capturing data.

begin-datetime
A date and time (timestamp) in the given form. For Extract, the timestamp value must be greater than the timestamp at which the Extract was registered with the database.

SOURCE URI *source_uri*
Specifies the source URI after the source keyword to indicate where the data is originated. The format of this URI contains the protocol (only supports trail), hostname, port number of the Distribution Service, and location of the source trail files.

TARGET URI *target_uri*
At least one TARGET option must be specified. It is treated as a complete object. If you need to specify something complicated, you need to use the REST API. Admin Client only supports a limited set of target settings and doesn't merge target setting with previous call.

CANONICAL
Defines a single byte order, a single floating-point representation of data.

TEXT
Text data.

SQL
Valid SQL statements.

XML
XML formatted data.

URI
Specifies the URI of the target distribution path.

FORMAT
Specifies the format of the URI of the target distribution path.

SIZEMB
Specifies the size of the trail sequence.

SEQLEN
Specifies the sequence length of the trail file.

PROXY URI
Specifies the proxy URI of the target distribution path.

TYPE
Specifies the type of connection between the source and target distribution path. You can choose a SOCKS PROXY or HTTP PROXY.

CSALIAS
Credential store alias used by the encryption profile for the distribution path.

CSDOMAIN
Domain of the credential store used by the encryption profile for the distribution path.

COMPRESSION
Specifies if the trail sequence has to be compressed. If you set it to YES, then you need to specify the threshold for the compression.

RULE

> **Note**
>
> From the Admin Client, if you create a rule while a previous exists, then the previous rule is removed and the new rule is added. You cannot append rules from the Admin Client. If you need to create complicated rules or append rules to existing rules then use the web interface or the REST API call directly.

At least one RULE option must be specified. For example:

```
ALTER DISTPATH dpe RULE FILTER CHUNKIDS (1, 2, 3)
```

PASSTHRU
See PASSTHRU | NOPASSTHRU in *Parameters and Functions Reference for Oracle GoldenGate*.

FILTER
At least one FILTER option must be specified. By default, the RELATION between the filters is OR and the action is KEEP.

CHUNKIDS *chunk_ids*
Specify a rule to filter records by their chunk ID (sharding). The list of chunk IDs must be parenthesized and comma separated.

OBJECTNAMES *object_names*
Specify a rule to filter records by their object name. The list of object names must be parenthesized and comma separated. An object name must follow the following grammar:

```
[cdb_name.]schema_name.table_name
```

For example:

```
ALTER DISTPATH dpe RULE FILTER OBJECTNAMES (hr.emp,
pdbnorth.hr.department)
```

OBJECTTYPES [DML | DDL | PROCEDURE]
Specify a rule to filter records by their object type. At least one object type must be specified.

TAGS *binary_tags*
Specify a rule to filter records by their tag. The list of tags must be parenthesized and comma separated. A tag must be a hexadecimal or binary value string and prefixed by the keywords HEXVALUE, HEXMASK, BINVALUE, and BINMASK. In case the tag is a BITMASK, the filter performs a bitwise AND operation between the mask and the tag value of an LCR record. If the result is equal to the MASK, then the action is applied. For example:

```
ALTER DISTPATH dpe RULE FILTER TAGS (hexvalue A4, hexvalue 18, hexmask
F0, hexvalue F8F, binvalue 01001100, binmask 0110)
```

PROCEDUREFEATURENAMES *feature_names*
Specify a rule to filter records by procedure feature name. The list of procedure feature names must be parenthesized and comma separated. For example:

```
ALTER DISTPATH dpe RULE FILTER PROCEDUREFEATURENAMES (RAS, AUTOCDR, AQ)
```

COLUMNVALUES *column_values*
Specify a rule to filter records by their column value. The filtering rules must follow this grammar and be comma separated:

```
[cdb_name.]schema_name.table_name.column_name ( EQ | NE | LT | GT | LE |
GE ) column_value [ BEFORE | AFTER ]
```

EQ = equal
NE = not equal
LT = less than
GT = greater than
LE = less or equal
GE = grater or equal
For example:

```
ALTER DISTPATH dpe RULE FILTER COLUMNVALUES
(pdbnorth.ggadmin.hr.emp.emp_id EQ 0 BEFORE,
cdbsouth.c##ggadmin.hr.emp.emp_id GT 100)
```

OPTIONS
At least one option must be specified:

AUTORESTART {RETRIES *retries* | DELAY *delay*}
Specifies that the distribution path is automatically restarted, how many times to retry the start, and any delay.

CRTICAL [YES | NO]
Indicates that the distribution path is critical to the deployment. The default is NO.

EOFDELAY *eofdelay*
Specifies how often Extract, a data pump, or Replicat checks for new data after it has reached the end of the current data in its data source.

See Change the Path Filtering to know about the rules for filter settings for an existing path.

AUTHENTICATION OAUTH
Use this option if you are using external Identity Provider (IDCS) authorization profile. This will set up the flow from the Distribution Service to the Receiver Service.

> **Note**
>
> If your deployment is enabled for IDCS, you can still choose to authentication using other authentication options.

AUTHENTICATION CERTIFICATE *certificate-name*
Identifier of distribution path-specific client certificate uploaded and managed in Administration Service.

AUTHENTICATION USERIDALIAS
Alternative to certificate authentication, you can associate each distribution path with a target DBLOGIN USERIDALIAS.

ENCRYPTIONPROFILE
Specifies the name of the encryption profile for the distribution path.

Examples

```
ALTER DISTPATH dpe BEGIN NOW
```

```
ALTER DISTPATH dpe BEGIN SEQNO 1 RBA 10355
```

```
ALTER DISTPATH dpn OPTIONS AUTORESTART RETRIES 3
```

```
ALTER DISTPATH dpe RULE FILTER OBJECTNAMES (hr.*, sales.*) ACTION EXCLUDE
```

```
ALTER DISTPATH dpe RULE FILTER TAGS (AE00, MASK AB00, FF)
```

```
ALTER DISTPATH dpe RULE FILTER COLUMNVALUES (hr.employess.deptno NE nope,
hr.employees.employee_id
    EQ 3 AFTER, hr.employees.deptno GE 5, hr.employees.employee_id GT 5
BEFORE)
```

```
ALTER DISTPATH dpe RULE FILTER OBJECTNAMES(hr.*,sales.*) RELATION AND ACTION
EXCLUDE
```

This example excludes all partitions matching `part1*` from all tables:

```
ALTER DISTPATH dpe RULE FILTER PARTITIONNAMES (part_2022*) ACTION EXCLUDE
```

ALTER ENCRYPTIONPROFILE

Use `ALTER ENCRYPTIONPROIFLE` to change the encryption profile name and default settings.

Syntax

```
ALTER ENCRYPTIONPROFILE encryption-profile-name
     DEFAULT [ YES | NO ]
     | OCIKMS [ USER user-ocid ]
                 [ APIKEY api-key-file
                   FINGERPRINT fingerprint ] )
```

`APIKEY` is supplied when using OCI KMS. Also see ADD ENCRYPTIONPROFILE.

Example

```
ALTER ENCRYPTIONPROFILE apn OCIKMS USER
ocid1.user.oc1..bbbbbbbbccccccccccccddddddddddddddddddxxxxxxxxxxxxyyyyyyyyzzzz
      APIKEY /home/dba/keys/oci_api_key.pem
      FINGERPRINT 11:22:33:aa:bb:cc:dd:ee:bb:11:22:33:44:ee:ff:ff
```

ALTER EXTRACT

Use `ALTER EXTRACT` for the following purposes:

- To change the attributes of an Extract group created with the `ADD EXTRACT` command.
- To increment a trail to the next file in the sequence.
- To upgrade to an integrated capture configuration.
- To downgrade from an integrated capture configuration.
- To position any given Db2 for i journal at a specific journal sequence number.

Before using this command, stop the Extract with the `STOP EXTRACT` *`group_name`* command.

Syntax

```
ALTER EXTRACT group-name
         [, BEGIN {NOW | yyyy-mm-dd[ hh:mi:[ss[.cccccc]]]} |
|            [, EXTRBA archive-offset-number]
|            [, ADD_EXTRACT_attribute  | SCN value]
|            [, LSN
|            [, EOL
|            [, DESC
|            [ CRITICAL    [ YES | NO ]     ]
|            [ ENCRYPTIONPROFILE     encryption-profile-name]
|            [ PROFILE
|                [ AUTOSTART   [ YES | NO ]
|                  [ DELAY     delay-number   ] ]
|                  [ AUTORESTART [ YES | NO ]
|                       [ RETRIES          retries-number ]
|                       [ WAITSECONDS      wait-number    ]
|                       [ RESETSECONDS     reset-number   ]
|                       [ DISABLEONFAILURE [ YES | NO ] ] ] ]
|             [ LOGNUM lognum]
|             [ LOGPOS logpos
```

The following `ALTER EXTRACT` options are supported for Db2 for i to position Extract for a given journal:

```
ALTER EXTRACT {BEGIN {NOW | yyyy-mm-ddT[ hh:mi:[ss[.cccccc]]Z]}
[JOURNAL journal_library/journal_name [JRNRCV receiver_library/
  receiver_name]] |
, EOL [JOURNAL journal_library/journal_name
  [JRNRCV receiver_library/receiver_name]] |
, SEQNO sequence_number [JOURNAL journal_library/journal_name
  [JRNRCV receiver_library/receiver_name]]}
```

BEGIN {NOW | *yyyy-mm-ddT[hh:mi:[ss[.cccccc]]]Z}*
[JOURNAL journal_library/journal_name
[JRNRCV *receiver_library/ receiver_name*]] |
, EOL [JOURNAL *journal_library/journal_name*
[JRNRCV *receiver_library/receiver_name*]] |
, SEQNO *sequence_number* [JOURNAL *journal_library/journal_name*
[JRNRCV *receiver_library/receiver_name*]]

These IBM for i options allow journal-specific Extract positioning after the global start point is issued with `ADD EXTRACT`. A specific journal position set with `ALTER EXTRACT` does not affect any global position that was previously set with `ADD EXTRACT` or `ALTER EXTRACT`; however a global position set with `ALTER EXTRACT` overrides any specific journal positions that were previously set in the same Extract configuration.

> **Note**
>
> `SEQNO`, when used with a journal in `ALTER EXTRACT`, is the journal sequence number that is relative to that specific journal, not the system sequence number that is global across journals.

group-name
The name of the Extract group that is to be altered.

{BEGIN {NOW | T*yyyy-mm-dd[hh:mi:[ss[.cccccc]]]*Z}

NOW
For all databases except Db2 LUW, `NOW` specifies the time at which the `ALTER EXTRACT` command is issued.
`NOW` specifies the time at which the `ALTER EXTRACT` command is issued.
For Db2 LUW, only commit and end transaction records contain timestamps, so the Extract starting position can only be calculated relative to those timestamps. This is a limitation of the API that is used by Oracle GoldenGate. It must be noted that positioning by timestamp is not accurate and can also take a long time. It is recommended to use `LRI` or `EOL` options wherever possible.

YYYY-MM-DDThh:mm:ssZ
A date and time (timestamp) in the given form. For example, `2017-07-14T14:54:45Z`.

yyyy-mm-dd[hh:mi:[ss[.cccccc]]]
A date and time (timestamp) in the given form. For an Oracle Extract in integrated mode, the timestamp value must be greater than the timestamp at which the Extract was registered with the database.
Positioning by timestamp in a SQL Server transaction log is affected by the following characteristics of SQL Server:

- The timestamps recorded in the SQL Server transaction log use a 3.3333 microsecond (ms) granularity. This level of granularity may not allow positioning by time between two transactions, if the transactions began in the same 3.3333 ms time interval.
- Timestamps are not recorded in every SQL Server log record, but only in the records that begin and commit the transaction, as well as some others that do not contain data.

- SQL Server timestamps are not from the system clock, but instead are from an internal clock that is specific to the individual processors in use. This clock updates several times a second, but between updates it could get out of sync with the system clock. This further reduces the precision of positioning by time.
- Timestamps recorded for log backup files may not precisely correspond to times recorded inside the backup (however this imprecision is less than a second).

Positioning to an LSN is precise.
Positioning by timestamp in PostgreSQL includes the following scenarios:

- Scenario 1

  If `track_commit_timestamp` is `off`, the following output will be displayed when the Extract process starts irrespective of what positioning method is used:

  ```
  2020-04-29 02:15:54 INFO OGG-01517 Position of first record processed
  LSN: 0/2222C20, Jan 1, 1970 12:00:00 PM.
  ```

- Scenario 2

  If the `track_commit_timestamp` is enabled before Extract is registered then the correct timestamp will be displayed once the records are pushed in the source database as mentioned in the following example:

  ```
  2020-04-29 02:19:07 INFO OGG-01515 Positioning to begin time Apr
  29,2020 2:18:38 AM.
  ```

- Scenario 3

  If `track_commit_timestamp` is enabled after the Extract is registered, then there may be chances that the older records are available in the log for which the commit timestamp is not built up with the associated `transaction ID`. In that case, if Extract does not get the timestamp then it will fallback using the default timestamp mentioned in scenario 1. The output will be similar to the following:

  ```
  020-04-29 01:55:07 INFO OGG-01517 Position of first record processed
  LSN: 0/221D028, Jan 1, 1970 12:00:00 PM.
  ```

- Past timestamp cannot be specified if the replication slot has moved away.

START
Valid for PostgreSQL.
Adds an Extract without mentioning `BEGIN NOW` or `LSN`. Extract will start from the replication slot restart position automatically.

ADD_EXTRACT_*attribute*
You can change any of the attributes specified with the `ALTER EXTRACT` command, except for the following:

- Altering an Extract specified with the `EXTTRAILSOURCE` option.
- Altering the number of RAC threads specified with the `THREADS` option.

For these exceptions, delete the Extract group and then add it again.

If using the BEGIN option, do not combine other options in the statement. Issue separate statements, for example:

```
ALTER EXTRACT exte, BEGIN 2019-01-01
ALTER EXTRACT exte, ETROLLOVER
ALTER EXTRACT exte, SCN 789000
```

If using the SCN or BEGIN option for Integrated Extract, it requires a DBLOGIN, and the SCN or timestamp value specified cannot be below the outbound server's first SCN or timestamp. To find the outbound server's first SCN, issue the following command:

```
INFO EXTRACT group_name, SHOWCH DETAIL
```

The first SCN value is listed as shown in the following example:

```
Integrated Extract outbound server first scn: 0.665884 (665884)
```

EXTRBA *offset_number*
Valid for Db2 z/OS.
Specifies the relative byte address within a transaction log at which to begin capturing data. The required format is 0X*nnn*, where *nnn* is a 1 to 20 digit hexadecimal number (the first character is the digit zero, and the second character can be upper or lower case letter x).

EOL
Valid for Db2 for i, Db2 LUW, MySQL, PostgreSQL, and SQL Server.
Configures processing to start at the end of the log files (or journals) that the next record will be written to. Any active transactions will not be captured.
For Db2 LUW, it configures processing to start at the active LRI value in the log files. The active LRI is the position at the end of the log files that the next record will be written to. Any active transactions will not be captured.
For PostgreSQL, DBLOGIN is required for position by EOL.
For MySQL, it finds the position corresponding to the end of the file and starts reading transactions from there. The EOL position is not exact, if data is continuously written to the binary log.

TRANLOG LRI *LRI_number*
(Db2 LUW) You can use this option for Db2 LUW systems to specify the LRI record value for the checkpoint transaction log.
For PostgreSQL, DBLOGIN is required for position by EOL.

REPORT *file_name*
Specifies the full path name of an Extract report file in a location other than the default of dirrpt within the Oracle GoldenGate directory.

SCN *value*
Valid for Oracle.
Starts Extract at the transaction in the redo log that has the specified Oracle system change number (SCN). For Extract, the SCN value must be greater than the SCN at which the Extract was registered with the database. For SCN or BEGIN option, DBLOGIN is required, and the SCN or timestamp value specified cannot be below the outbound server's first SCN or timestamp.

LSN *value*
Valid for Db2 z/OS, PostgreSQL, and SQL Server.

Specifies the transaction LSN at which to start capturing data. An alias for this option is EXTLSN.
For Db2 z/OS, LSN value can be either a series of hex digits or a series of decimal digits. If it is in hex format, it must be prefixed with either 0X or 0x followed by 1 to 20 hex digits: 0-9, a-g, or A-G. If it is in decimal format, there is no prefix and there are 1-25 decimal digits 0-9.
For PostgreSQL, LSN value can be hi or lo. Set the value as hi for the entry point of the log file. Lo is the offset in the log file. The LSN position should lie between the replication slot restart position and write ahead log current location. If the position specified itself exists between the mentioned range then Extract will throw an error.
For SQL Server, LSN specifies the transaction LSN at which to start capturing data. An alias for this option is EXTLSN.
The specified LSN should exist as a valid tran_begin_lsn found in the cdc.lsn_time_mapping system table, otherwise the Extract will attempt to position after the provided LSN value.
Valid LSN specifications for SQL Server consists of the following:

- Colon separated hex string (8:8:4) padded with leading zeroes and 0X prefix, as in 0X00000d7e:0000036b:0001
- Colon separated hex string with 0X prefix and without leading zeroes, as in 0Xd7e:36b:1

You can find the minimum LSN available by querying the following:

```
SELECT min([tran_begin_lsn]) FROM [cdc].[lsn_time_mapping] with (nolock)
where tran_id <> 0x00
```

The following examples show the use of LSN value for Db2 z/OS:
Example 1:

```
ALTER EXTRACT extn TRANLOG, LSN 0xDEE40E4F27A3245400
```

Example 2:

```
ALTER EXTRACT extn TRANLOG, LSN 22216433159121980904448
```

The following example shows the LSN value for SQL Server:

```
ALTER EXTRACT extn TRANLOG, LSN 0X00000d7e:0000036b:0001
```

LOGNUM
Valid for MySQL.
This is the log file number. For example, if the required log file name is test.000034, the LOGNUM value is 34. Extract will search for this log file.

> **Note**
>
> In Microservices Architecture, ALTER EXTRACT will fail if the LOGNUM value contains zeroes preceding the value. For example, ALTER EXTRACT ext1, TRANLOG, LOGNUM 000001, LOGPOS 0 will fail. Instead, set LOGNUM to 1 for this example to succeed.

LOGPOS
Valid for MySQL.

This is an event offset value within the log file that identifies a specific transaction record. Event offset values are stored in the header section of a log record. To position at the beginning of a `binlog` file, set the `LOGPOS` as 0.

`ETROLLOVER`
Use for manual recovery situations that require repositioning and regenerating trail files for a primary Extract and when upgrading Oracle GoldenGate from a previous version.
Causes Extract to create a new incarnation of the trail file and increments to the next file in the trail sequence when restarting, requiring readers such as pump or Replicat, to be manually repositioned to the new trail sequence number.
With `ETROLLOVER`, Extract captures all records after the specified SCN with which Extract is starting irrespective of whether it was already captured and written to trail or not. Without `ETROLLOVER`, Extract will skip the capture of any records that are already captured and written to trail.
From 19c onwards, during Distribution Service processing:

- If the primary Extract on the source deployment is upgraded with target trail file `ETROLLOVER`, then theDistribution Service automatically detects the source trail file `ETROLLOVER` and starts reading from the next input trail file. This is the same command as data pump `ALTER EXTRACT` *`groupname`* `EXTSEQNO` *`seqno+1`* `EXTRBA 0`.
- After upgrading the Distribution Service, it automatically performs the `ETROLLOVER` for the output trail file upon restarting and writes the next trail file properly. This is the same as data pump `ALTER EXTRACT` *`groupname`* `ETROLLOVER`.

`DESC '`*`description`*`'`
Specifies a description of the group, such as `'Extracts account_tab on Serv1'`. Enclose the description within single quotes. You may use the abbreviated keyword `DESC` or the full word `DESCRIPTION`.

`ENCRYPTIONPROFILE`
Specifies the name of the encryption profile for the Extract. This name is case sensitive so you must use the exact name that you entered with `ADD EXTRACT`.

`GTIDSET` ***`gtidset`***
Valid for MySQL.
Specifies the initial positioning of of Extract by using the position type positiob ing the GTID set option for GTID-based capture for MySQL. The supported MySQL sources for GTID set are MySQL Server 8.0, MySQL Server 5.7, MySQL Database Service (MDS). The maximum supported GTID set size is 64 KB.
Syntax:
`ALTER EXTRACT` *`extract_name`*`, TRANLOG, GTIDSET` *`gtidset`*

The following examples include all the combinations of valid formats of GTID set.
Example 1:

```
ALTER EXTRACT exte, TRANLOG, GTIDSET "E11FA47-71CA-11E1-9E33-C80AA9429562:4"
```

Example 2:

```
ALTER EXTRACT exts, TRANLOG, GTIDSET "3E11FA47-71CA-11E1-9E33-
C80AA9429562:1-10"
```

Example 3:

```
ALTER EXTRACT extn, TRANLOG, GTIDSET "3E11FA47-71CA-11E1-9E33-
C80AA9429562:1-3:11:47-49"
```

Example 4:

```
ALTER EXTRACT extw, TRANLOG, GTIDSET "2174B383-5441-11E8-B90A-
C80AA9429562:1-3,24DA167-0C0C-11E8-8442-00059A3C7B00:1-19"
```

> **Note**
>
> As shown in example 4, when GTID set contains multiple uuids, to form a REST API request, this GTID set is broken into multiple GTID sets using comma separator.

CRITICAL
Indicates if the process is critical for the deployment.

PROFILE
Name of the auto start profile. This name is case sensitive so you must use the exact name that you entered with `ALTER EXTRACT`.

AUTOSTART
Specifies whether the managed process has to be started automatically when the Administration Service starts. The default value is `YES`.

RETRIES
The maximum number of tries for restarting the task before canceling retry efforts. This is optional.

WAITSECONDS
The duration (in seconds) in which the retries are counted.

RESETSECONDS
Resets the duration in which the retries are counted.

DISABLEONFAILURE
If set to `TRUE`, then the task is disabled when the number of retries is exhausted.

INFO EXTRACT *group-name*, SHOWCH DETAIL
The first SCN value is listed as shown in the following example:

```
Integrated Extract outbound server first scn: 0.665884 (665884)
```

Examples

The following alters Extract to start processing data from January 1, 2019.

```
ALTER EXTRACT exte, BEGIN 2019-01-01
```

The following alters Extract to start processing at a specific location in the trail.

```
ALTER EXTRACT exte, EXTSEQNO 26, EXTRBA 338
```

The following alters Extract in a SQL Server environment to start at a specific LSN.

```
ALTER EXTRACT exte, LSN 0Xd7e:36b:1
```

The following alters Extract to increment to the next file in the trail sequence.

```
ALTER EXTRACT exte, ETROLLOVER
```

The following alters Extract to upgrade to integrated capture.

```
ALTER EXTRACT exte, UPGRADE INTEGRATED TRANLOG
```

The following command uses `ETROLLOVER` to **prevent filtering of duplicate records** with backward SCN 778899.

```
ALTER EXTRACT exte, SCN 778899
ALTER EXTRACT exte ETROLLOVER
START EXTRACT exte
```

If you do not use `ETROLLOVER`, then if you alter Extract to the previous SCN value, it will not process duplicate records. The command to alter Extract to **filter duplicate records** is as follows:

```
ALTER EXTRACT exte, SCN 778899
START EXTRACT exte
```

The following shows `ALTER EXTRACT` for an IBM for i journal start point.

```
ALTER EXTRACT exte, SEQNO 1234  JOURNAL accts/acctsjrn
```

The following shows `ALTER EXTRACT` for an IBM for i journal and receiver start point.

```
ALTER EXTRACT exte, SEQNO 1234 JOURNAL accts/acctsjrn JRNRCV accts/jrnrcv0005
```

The following example alters an Extract on a Db2 LUW system.

```
ALTER EXTRACT exte, TRANLOG LRI 8066.322711
```

The following example shows the ALTER EXTRACT options used with PostgreSQL:

```
ALTER EXTRACT exte, EOL
ALTER EXTRACT POSTEXT, LSN 0/156784
ALTER EXTRACT POSTEXT, BEGIN 2022-02-18T14:50:43.4230Z
```

ALTER EXTTRAIL

Use ALTER EXTTRAIL to change the attributes of a trail that was created with the ADD EXTTRAIL command (a trail on the local system). The change takes effect the next time that Extract starts.

Before using this command, stop the Extract using the STOP EXTRACT *group_name* command.

Syntax

```
ALTER EXTTRAIL trail_name, EXTRACT group_name
[, MEGABYTES n]
```

trail_name
The relative or fully qualified path name of the trail.

group_name
The name of the Extract group to which the trail is bound.

MEGABYTES *n*
The maximum size of a file, in megabytes. The default is 500. After using this option, issue the SEND EXTRACT command with the ROLLOVER option to close the current trail file and open a new one.

Examples

```
ALTER EXTTRAIL north/ea, EXTRACT exte,  MEGABYTES 200
```

ALTER RECVPATH

Use ALTER RECVPATH to change the attributes of target-initiated distribution path in the Receiver Service.

Syntax

```
ALTER RECVPATH path-name
  |         ( BEGIN   ( NOW
  |                   | SEQNO trail-sequence-number RBA relative-byte-address
  |                   | begin-datetime )
  |         | SOURCE  ( URI           source-uri
  |                   | PROXY URI     proxy-uri )
  |         | TARGET  ( FORMAT        ( CANONICAL | TEXT | SQL | XML )
  |                   | SIZEMB        megabytes-number
  |                   | SEQLEN        sequence-length
  |                           TYPE    ( SOCKS | HTTP )
  |                           [ CSALIAS credential-store-alias
  |                             [ CSDOMAIN credential-store-domain ] ]
```

```
  |                   | ENCRYPTIONPROFILE encryptionprofile_name
  |                   | COMPRESSION ( ON | OFF | THRESHOLD compression-
threshold ) )
  |         | RULE    ( PASSTHRU
  |                   | FILTER [ CHUNKIDS    chunk-ids ]
  |                            [ OBJECTNAMES object-names-wildcard ]
  |                            [ OBJECTTYPES [ DML ] [ DDL ] [ PROCEDURE ] ]
  |                            [ TAGS        binary-tags ]
  |                            [ PROCEDUREFEATURENAMES feature-names-
wildcard ]
  |                            [ COLUMNVALUES column-values ]
  |                            [ PARTITIONNAMES partition-names-wildcard ] )
  |                   [ RELATION    ( AND  | OR  ) ]
  |                   [ ACTION      ( INCLUDE | EXCLUDE ) ]
  |         | OPTIONS [ AUTORESTART RETRIES retries [ DELAY delay ] ]
  |                   [ CRITICAL ( YES | NO ) ]
  |                   [ EOFDELAY    eofdelay         ]
  |         | AUTHENTICATION ( OAUTH
                                 | ( CERTIFICATE certificate-name )
                                 | ( USERIDALIAS alias [DOMAIN domain
                                 |  (ENCRYPTIONPROFILE encryption-profile-
name] ) ) )
```

path_name
The name of the distribution path you want to change.

BEGIN {NOW | SEQNO *sequence_number* RBA *relative_byte_address* | *begin_datetime*}
Specifies a timestamp in the data source at which to begin processing.

NOW
Specifies the time at which the `ADD EXTRACT` command is issued.

sequence_number relative_byte_address
The sequence number of an Oracle redo log and RBA within that log at which to begin capturing data.

begin_datetime
A date and time (timestamp) in the given form. For an Extract in integrated mode, the timestamp value must be greater than the timestamp at which the Extract was registered with the database.

SOURCE URI *source_uri*
Specifies the source URI after the source keyword to indicate where the data is originated. The format of this URI contains the protocol (only supports trail), hostname, port number of the Receiver Service, and location of the source trail files.

PROXY URI *proxy_uri*
Specifies the proxy URI after the proxy keyword to indicate where the data is originated. The format of this URI contains the protocol (only supports trail), hostname, port number of the Receiver Service, and location of the source trail files.

TARGET
At least one `TARGET` option must be specified. It is treated as a complete object. If you need to specify something complicated, you need to use the REST API. Admin Client only supports a limited set of target settings and doesn't merge target setting with previous call.

CANONICAL
Defines a single byte order, a single floating-point representation of data.

TEXT
Text data.

SQL
Valid SQL statements.

XML
XML formatted data.

FORMAT
Specifies the format of the URI of the target distribution path.

SIZEMB *megabytes_number*
Sets the size of the distribution path in megabytes.

SEQLEN *sequence_length*
Sets the sequence length of the distribution path.

TYPE
Sets the proxy type that the distribution path uses, HTTP or SOCKS.

CSALIAS *credential_store_alias*
Specifies your credential store alias name.

CSALIAS *credential_store_domain*
Specifies your credential store domain name.

AUTHENTICATION OAUTH
Use this option if you are using external Identity Provider (IDCS) authorization profile. This will set up the flow from the Distribution Service to the Receiver Service.

> **Note**
>
> If your deployment is enabled for IDCS, you can still choose to authentication using other authentication options.

AUTHENTICATION CERTIFICATE *certificate-name*
Identifier of distribution path-specific client certificate uploaded and managed in Administration Service.

AUTHENTICATION USERIDALIAS
Alternative to certificate authentication, you can associate each distribution path with a target `DBLOGIN USERIDALIAS`.

ENCRYPTIONPROFILE
Specifies the name of the encryption profile for the distribution path.

COMPRESSION {ON | OFF | THRESHOLD *compression_threshold*}
Specifies whether your data is compressed or not. If set to `ON`, then you can specify the threshold level.

RULE
At least one RULE option must be specified. For example:

```
ALTER RECVPATH rcvp RULE FILTER CHUNKIDS (1, 2, 3)
```

PASSTHRU
See PASSTHRU | NOPASSTHRU in *Parameters and Functions Reference for Oracle GoldenGate*.

FILTER
At least one FILTER option must be specified. By default the RELATION between the filters is OR and the action is KEEP.

CHUNKIDS *chunk_ids*
Specify a rule to filter records by their chunk ID (sharding). The list of chunk IDs must be parenthesized and comma separated.

OBJECTNAMES *object_names*
Specify a rule to filter records by their object name. The list of object names must be parenthesized and comma separated. An object name must follow the following grammar:

```
[cdb_name.]schema_name.table_name
```

For example:

```
ALTER RECVPATH rcvp RULE FILTER OBJECTNAMES (hr.emp,
pdbnorth.hr.department)
```

OBJECTTYPES [DML | DDL | PROCEDURE]
Specify a rule to filter records by their object type. At least one object type must be specified.

TAGS *binary_tags*
Specify a rule to filter records by their tag. The list of tags must be parenthesized and comma separated. A tag must be a hexadecimal or binary value string and prefixed by the keywords HEXVALUE, HEXMASK, BINVALUE, and BINMASK. In case the tag is a BITMASK, the filter performs a bitwise AND operation between the mask and the tag value of an LCR record. If the result is equal to the MASK, then the action is applied. For example:

```
ALTER RECVPATH rcvp RULE FILTER TAGS (hexvalue A4, hexvalue 18, hexmask
F0, hexvalue F8F, binvalue 01001100, binmask 0110)
```

PROCEDUREFEATURENAMES *feature_names*
Specify a rule to filter records by procedure feature name. The list of procedure feature names must be parenthesized and comma separated. For example:

```
ALTER RECVPATH rcvp RULE FILTER PROCEDUREFEATURENAMES (RAS, AUTOCDR, AQ)
```

COLUMNVALUES *column_values*
Specify a rule to filter records by their column value. The filtering rules must follow this grammar and be comma separated:

```
[cdb_name.]schema_name.table_name.column_name ( EQ | NE | LT | GT | LE |
GE ) column_value [ BEFORE | AFTER ]
```

`EQ` = equal
`NE` = not equal
`LT` = less than
`GT` = greater than
`LE` = less or equal
`GE` = grater or equal
For example:

```
ALTER RECVPATH rcvp RULE FILTER COLUMNVALUES
(pdbnorth.ggadmin.hr.emp.emp_id EQ 0 BEFORE,
cdbsouth.c##ggadmin.hr.emp.emp_id GT 100)
```

OPTIONS
At least one option must be specified:

AUTORESTART {RETRIES *retries* | DELAY *delay*}
Specifies that the distribution path is automatically restarted, how many times to retry the start, and any delay.

CRTICAL [YES | NO]
Indicates that the distribution path is critical to the deployment. The default is `NO`.

EOFDELAY *eofdelay*
Specifies how often Extract, a data pump, or Replicat checks for new data after it has reached the end of the current data in its data source.

ENCRYPTIONPROFILE
Specifies the name of the encryption profile for the Receiver path.

Examples

```
ALTER RECVPATH rcvp BEGIN NOW
```

```
ALTER RECVPATH rcvp BEGIN SEQNO 1 RBA 10355
```

```
ALTER RECVPATH rcvp OPTIONS AUTORESTART RETRIES 3
```

```
ALTER RECVPATH rcvp RULE FILTER OBJECTNAMES (hr.*, sales.*) ACTION EXCLUDE
```

```
ALTER RECVPATH rcvp RULE FILTER TAGS (AE00, MASK AB00, FF)
```

```
ALTER RECVPATH rcvp RULE FILTER COLUMNVALUES (c##ggadmin.hr.employees NE
nope, c##ggadmin.hr.employees
    EQ 3 AFTER, c##ggadmin.hr.employees GE 5, c##ggadmin.hr.employees GT 5
BEFORE)
```

```
ALTER RECVPATH rcvp RULE FILTER OBJECTNAMES(hr.*,sales.*) RELATION AND ACTION
EXCLUDE
```

ALTER RMTTRAIL

Use `ALTER RMTTRAIL` to change the attributes of a trail that was created with the `ADD RMTTRAIL` command (a trail on a remote system). The change takes effect the next time that Extract starts.

Syntax

```
ALTER RMTTRAIL trail_name, EXTRACT group_name
[, MEGABYTES n]
```

trail_name
The relative or fully qualified path name of the trail. For example, `dirdat\ea`.

group_name
The name of the Extract group to which the trail is bound.

`MEGABYTES n`
The maximum size of a file, in megabytes. The default is 500. After using this option, issue the `SEND EXTRACT` command with the `ROLLOVER` option to close the current trail file and open a new one.

Example

```
ALTER RMTTRAIL north/ea, EXTRACT exte, MEGABYTES 200
```

ALLOWNESTED

Use the `ALLOWNESTED` and `NOALLOWNESTED` commands to enable or disable the use of nested `OBEY` files. A nested `OBEY` file is one that contains another `OBEY` file, see OBEY.

Syntax

```
ALLOWNESTED | NOALLOWNESTED
```

`ALLOWNESTED`
Enables the use of nested `OBEY` files. There is no maximum of the number of nested files.

NOALLOWNESTED
This is the default setting. If you try to run a nested obey file, then it displays the following error

```
Nested OBEY scripts not allowed. Use ALLOWNESTED to allow nested scripts.
```

The following example illustrates a nested `OBEY` file. Assume an `OBEY` file named `addcmds.txt`. Inside this file, there is another `OBEY` command that calls the `OBEY` file named `startcmds.txt`, which executes another set of commands.
The following example creates an OBEY file (`adder.oby`) to add Extract, Replicat, and trail files and includes another obey file (startcmds.oby).

```
vi ADDER.oby
DBLOGIN USERIDALIAS ggeast
ADD EXTRACT exte, TRANLOG, BEGIN NOW
ADD EXTTRAIL east/ea, EXTRACT exte
ADD EXTRACT extw, TRANLOG, BEGIN NOW
ADD REPLICAT repw, EXTTRAIL west/ew, BEGIN NOW
OBEY startcmds.oby
```

The `startcmds.oby` file contains the following:

```
START EXTRACT *
INFO EXTRACT *, DETAIL
START REPLICAT *
INFO REPLICAT *, DETAIL
```

To exectue these obey commands from the command line, you need to use the ALLOWNESTED command:

```
ALLOWNESTED
OBEY adder.oby
```

ALTER HEARTBEATTABLE

Use `ALTER HEARTBEATTABLE` to alter existing seed, heartbeat, and history table options that you set with `ADD HEARTBEATTABLE`.

This command requires a `DBLOGIN`. On a CDB database, a PDB login is required.

Valid for Oracle, Db2 z/OS, Db2 LUW, Db2 for i, MySQL, PostgreSQL, SQL Server, and Sybase. This command is not valid for TimesTen and Teradata.

Syntax

```
ALTER HEARTBEATTABLE
[, FREQUENCY number_in_seconds]
[, RETENTION_TIME number_in_days] |
[, PURGE_FREQUENCY number_in_days]
[, TARGETONLY | NOTARGETONLY]
```

FREQUENCY *number_in_seconds*
Specifies how frequently heartbeat records are generated. The default is 60 seconds.
Consider the following limits:

- For Oracle Database, the minimum value is 0 and the maximum is 7999.
- For DB2 for i Series, the minimum value is 0 and the maximum is 7999.
- For DB2 LUW and DB2 z/OS, the minimum value is 60 and the maximum is 7999.
- The frequency for DB2 z/OS and DB2 LUW must be a multiple of 60 for values less than 3600 and multiples for 3600 for values greater or equal to 3600.
- For MySQL, the minimum value is 0 and the maximum is 7999.
- For SQL Server, the minimum value is 10 and the maximum is 7999. It supports 0, which disables the SQL Server Agent Heartbeat table `UPDATE` job, but can only be set with `ALTER HEARTBEATTABLE`.
- For PostgreSQL, the minimum value is 60 and the maximum is 7999.
- For Sybase, the minimum value is 1 minute (60 seconds) and maximum is 133 minutes (7999 seconds).
- Databases that support setting `FREQUENCY` to 0 will pause the heartbeat record scheduler.

RETENTION_TIME
Specifies that heartbeat entries older than the retention time in the heartbeat history table are purged. The default is 30 days.
The minimum value for all databases is 1 and the maximum is 2147483646.

PURGE_FREQUENCY
Specifies how often the purge scheduler is run to delete table entries that are older than the retention time from the heartbeat history table. The default is 1 day.
For DB2 LUW and DB2 z/OS, the minimum value is 1 and the maximum is 31.
For all other supported databases, the minimum value is 1 and the maximum value is 199.

TARGETONLY | NOTARGETONLY
Valid for Oracle Database, DB2 for i Series, DB2 LUW, DB2 z/OS, MySQL, PostgreSQL, and SQL Server.
`TARGETONLY` modifies existing heartbeat seed and heartbeat tables by disabling supplemental logging on both tables. It drops the existing scheduler job for updating the heartbeat table.
`NOTARGETONLY` modifies existing heartbeat seed and heartbeat tables by enabling supplemental logging on both tables. It creates a new scheduler job for updating the heartbeat table.

Examples

```
ALTER HEARTBEATTABLE FREQUENCY 60


ALTER HEARTBEATTABLE RETENTION_TIME 30


ALTER HEARTBEATTABLE PURGE_FREQUENCY 1


ALTER HEARTBEATTABLE NOTARGETONLY
```

ALTER REPLICAT

Use `ALTER REPLICAT` to change the attributes of a Replicat group that was created with the `ADD REPLICAT` command. Before using this command, stop Replicat by issuing the `STOP REPLICAT` command.

> **Note**
>
> `ALTER REPLICAT` does not support switching from regular Replicat mode to coordinated mode. You must stop processes, make certain all of the en route data is applied to the target, roll the trail to a new trail, drop and create the Replicat group again, in coordinated mode. Then restart the processes.

Syntax

```
 ALTER REPLICAT
          group-name|
           [, INTEGRATED] | NONINTEGRATED ]
          option [, . . .]
         [, PARAMS file_name]
         [, REPORT file_name]
         [, BEGIN           {NOW | yyyy-mm-dd[ hh:mi:
[ss[.cccccc]]]}]
         [, EXTSEQNO        trail-sequence-number, EXTRBA trail-offset-
number ]
         [, EXTRBA rba
         [, DESC            description ]|
         [, ENCRYPTIONPROFILE encryption-profile-name ]|
         [, CRITICAL        [ YES | NO ]  ]|
         [, PROFILE         profile-name|
         [, AUTOSTART       [ YES | NO ]|
         [, DELAY           delay-number ] ]|
         [, AUTORESTART     [ YES | NO ]|
         [, RETRIES          retries-number ]|
         [, WAITSECONDS      wait-number    ]|
         [, RESETSECONDS     reset-number   ]|
         [, DISABLEONFAILURE [ YES | NO ]     ] ] ]
}
```

group_name
The name of the Replicat group that is to be altered.

INTEGRATED
Switches Replicat from non-integrated mode to integrated mode. Transactions currently in process are applied before the switch is made, see Switching from Non-Integrated Replicat to Parallel Non-Integrated Replicat.

NONINTEGRATED, CHECKPOINTTABLE *owner.table*
(Oracle) Switches Replicat from integrated mode to non-integrated mode.
For `CHECKPOINTTABLE`, specify the owner and name of a checkpoint table. This table must be created with the `ADD CHECKPOINTTABLE` command before issuing `ALTER EXTRACT` with

NONINTEGRATED, see Select a Replicat Type for your Deployment for more information about integrated Replicat.

***option* [, . . .]**
Use this to change any description or service option that was configured with the ADD REPLICAT command, except for the CHECKPOINTTABLE and NODBCHECKPOINT options. There is no option to alter a Replicat with CHECKPOINTTABLE to replicate with or without CHECKPOINTTABLE. However, if you are switching from integrated to non-integrated Replicat, you can use a non-integrated Replicat without a checkpoint table, as shown in the following example:

```
ADD REPLICAT erep, INTEGRATED, EXTTRAIL ea, ggadmin.ggs_checkpoint


ALTER REPLICAT erep, NONINTEGRATED, CHECKPOINTTABLE ggadmin.ggs_checkpoint
```

BEGIN {NOW | *yyyy-mm-dd[hh:mm[:ss[.ccccccc]]]}*
Defines an initial checkpoint in the trail.

NOW
Begins replicating changes from the time when the group is created.

yyyy-mm-dd[hh:mm[:ss[.ccccccc]]]
Begins extracting changes from a specific time.

EXTSEQNO *sequence_number*
Specifies the sequence number of the file in a trail in which to begin processing data. Specify the sequence number, but not any zeroes used for padding. For example, if the trail file is \aa000000026, you would specify EXTSEQNO 26. If not specified, the default value is zero. By default, processing begins at the beginning of a trail unless this option is used. To use EXTSEQNO, you must also use EXTRBA. Contact Oracle Support before using this option.

EXTRBA *rba*
Valid for Db2 z/OS.
Specifies the relative byte address within the trail file that is specified by EXTSEQNO. Contact Oracle Support before using this option.

PARAMS *file_name*
Specifies a parameter file in a location other than the default of dirprm within the Oracle GoldenGate directory. Specify the fully qualified path name.

REPORT *file_name*
Specifies the full path name of a process report file in a location other than the default of dirrpt within the Oracle GoldenGate directory.

DESC '*description*'
Specifies a description of the group, such as 'Loads account_tab on Serv2'. Enclose the description within quotes.

ENCRYPTIONPROFILE
Specifies the name of the encryption profile for the Replicat.

CRITICAL
Indicates if the process is critical for the deployment.

PROFILE
There are options to select the Default or Custom profiles, if you've created one using the Profile page in Distribution Service.

AUTOSTART
Select this option to start the process when Administration Service starts.

DELAY
Time to wait in seconds before starting the process.

AUTORESTART
Controls how the process will be restarted if it terminates.

RETRIES
The maximum number of the task should be restarted before canceling retry efforts. This is optional.

WAITSECONDS
Specifies the time to wait before performing the retries.

RESETSECONDS
Resets the time for waiting.for retries.

DISABLEONFAILURE
If this option is enabled, then the task is disabled when the number of retries is exhausted.

Examples

```
ALTER REPLICAT erep, EXTSEQNO 53
```

```
ALTER REPLICAT erep, EXTSEQNO 53, EXTRBA 0
```

```
ALTER REPLICAT erep, BEGIN 2011-01-07 08:00:00
```

```
ALTER REPLICAT erep, INTEGRATED
```

```
ALTER REPLICAT erep, NONINTEGRATED, CHECKPOINTTABLE ggadmin.ggs_checkpoint
```

```
ALTER REPLICAT wrep, EXTSEQNO 53
```

CD

Use `CD` to change the Admin Client working directory.

Syntax

```
CD directory-name
```

`directory-name`

The name of the directory.

CLEAR INSTANTIATION CSN

Use `CLEAR INSTANTIATION CSN` on your target database to clear (reverse) the instantiation CSN manually. This command requires `DBLOGIN` where the user is the default Oracle GoldenGate schema.

Syntax

```
CLEAR INSTANTIATION CSN FOR [schema.]table FROM source-database-name
```

[schema.]table
The name of the table to clear the instantiation CSN on. If no schema is provided, the `DBLOGIN` user will be used.

source-database-name
The global name of the source database for which this is a target.

Example

```
CLEAR INSTANTIATION CSN FOR hr.employees FROM pdbeast.com
```

CLEANUP CHECKPOINTTABLE

Not valid for Replicat for Java, Oracle GoldenGate Applications Adapter, or Oracle GoldenGate Big Data.

Use `CLEANUP CHECKPOINTTABLE` to remove checkpoint records from the checkpoint table when there is no checkpoint file associated with it in the working Oracle GoldenGate directory. This command should only be used on Replicats that have been deleted. The purpose of this command is to remove checkpoint records that are not needed any more, either because groups were changed or files were moved.

Use the `DBLOGIN` command to establish a database connection before using this command.

Syntax

```
CLEANUP CHECKPOINTTABLE [[container. | catalog.]owner.table]
```

container. | catalog.
The Oracle pluggable database, if applicable. If this option is omitted, the catalog or pluggable database defaults to the one that is associated with the `SOURCEDB`, `USERID`, or `USERIDALIAS` portion of the `DBLOGIN` command (depending on the database).

owner.table
The owner and name of the checkpoint table to be cleaned up. If an owner and name are not specified, the table that is affected is the one specified with the `CHECKPOINTTABLE` parameter in the `GLOBALS` parameter file.

Example

```
CLEANUP CHECKPOINTTABLE ggadmin.ggs_checkpoint
```

CLEANUP EXTRACT

Use `CLEANUP EXTRACT` to delete run history for the specified Extract group. The cleanup keeps the last run record intact so that Extract can resume processing from where it left off. Before using this command, stop Extract by issuing the `STOP EXTRACT` command.

Syntax

```
CLEANUP EXTRACT group_name [, SAVE count]
```

group_name
The name of an Extract group or a wildcard (*) to specify multiple groups. For example, `T*` cleans up all Extract groups whose names start with T.

`SAVE` *`count`*
Excludes the specified number of the most recent records from the cleanup.

Examples

Example 1
The following deletes all but the last record.

```
CLEANUP EXTRACT exte
```

Example 2
The following deletes all but the most recent five records.

```
CLEANUP EXTRACT *, SAVE 5
```

CLEANUP REPLICAT

Use `CLEANUP REPLICAT` to delete run history for a specified Replicat group. The cleanup keeps the last run record intact so that Replicat can resume processing from where it left off.

Before using this command, stop Replicat by issuing the `STOP REPLICAT` command.

Syntax

```
CLEANUP REPLICAT group_name[threadID] [, SAVE count]
```

***group_name*[*threadID*]**
One of the following:

- *group_name*: The name of a Replicat group or a wildcard (*) to specify multiple groups. For example, `T*` cleans up all Replicat groups whose names begin with T. If the specified group (or groups) is a coordinated Replicat, the cleanup applies to all threads.
- *group_namethreadID*: A thread of a coordinated Replicat, identified by its full name (group name plus *threadID*), such as `finance003`.

SAVE *count*
Excludes the specified number of the most recent records from the cleanup.

Examples

The following deletes all but the last record.

```
CLEANUP REPLICAT repe
```

The following deletes all but the most recent five records.

```
CLEANUP REPLICAT *, SAVE 5
```

The following deletes all but the most recent five records for thread three of coordinated Replicat group `fin`.

```
CLEANUP REPLICAT repe, SAVE 5
```

CONNECT

You can use this command to connect to a Service Manager as well as the Administration Service URI. You must connect to your Service Manager before you can execute most of the Admin Client commands.

Admin Client allows connections when the server uses a self-signed certificate though this is not the default. Admin Client does not allow connecting to a server through HTTPS when the self-signed certificate is invalid. To override this behavior, use the `!` modifier with the `CONNECT` command.

For example, when using the Admin Client to connect to the Oracle GoldenGate Microservices Architecture services that are secured with a self-signed SSL certificate, you must use a command with the `!` modifier:

```
CONNECT https://myserver.example.org as oggadmin !
```

When using Admin Client to connect to an IDP-enabled deployment, use the `CONNECT` command with the `access-token` parameter.

Syntax

```
CONNECT server-url deployment-name access-token proxy-uri user-name password
```

server-url
The URL of the Service Manager or Administration Service that you want to connect to.

deployment-name
The name of the deployment that you want to connect to on the specified Service Manager. If only one deployment (except for Service Manager) is defined, that deployment is the default. Otherwise, there is not a default deployment and the DEPLOYMENT deployment-name option must be used.

access-token
Access token value to connect to the IDP-enabled deployment.

proxy-uri
The URI of your proxy server in the *schema*://*hostname*[:*port-number*] format.

user-name
The user name for the specified Service Manager.

password
The password for the specified user name. If you do not specify the password, you are prompted for it.

Example

```
CONNECT http://prodserver.mysite.com:9700 deployment Atlanta_1 as oggadmin
password welcome1
```

DBLOGIN USERIDALIAS

Use DBLOGIN to establish a database connection from the Admin Client. The user requires database privileges to be able to use the DBLOGIN command. Any other special privileges that are required for the Admin Client commands are listed with the reference documentation for that command.

> **Note**
>
> DBLOGIN doesn't work in Oracle GoldenGate 21c with Oracle Database19c database on OEL 7.9.

Syntax

```
DBLOGIN USERIDALIAS alias
```

alias
Specifies the alias of a database user credential that is stored in the Oracle GoldenGate credential store. This alias is the TNS connection identifier used to connect to a database.

To log into a pluggable database in an Oracle multitenant container database, the user must be stored as a connect string, such as `ggadmin@orcl`. To log into the root container, the user must be stored as a common user, including the `c##` prefix, such as `c##ggadmin@cdb$root`.

`DOMAIN` *`domain`*
Specifies the credential store domain for the specified alias. A valid domain entry must exist in the credential store for the specified alias. The default domain is `OracleGoldenGate`.

Examples

```
DBLOGIN USERIDALIAS ggeast
```

```
DBLOGIN USERIDALIAS ggeast DOMAIN OracleGoldenGate
```

DELETE AUTHORIZATIONPROFILE

Use this command to delete an authorization profile. However, the `localCredentialStore` profile, or the profile that is currently set for the deployment, cannot be deleted. When a profile is successfully deleted, information stored in the configuration file and securely stored information is removed.

Syntax

```
DELETE AUTHORIZATIONPROFILE profile-name
DEPLOYMENT deployment-name
```

`profile-name`
Specify the name of the authorization profile that you need to delete.

`deployment-name`
Specify the name of the deployment associated with the authorization profile.

Example

The following shows how to delete a profile:

```
DELETE AUTHORIZATIONPROFILE testProfile DEPLOYMENT ServiceManager
```

DELETE CHECKPOINTTABLE

Not valid for Replicat for Java, Oracle GoldenGate Applications Adapter, or Oracle GoldenGate Big Data.

Use `DELETE CHECKPOINTTABLE` to drop a checkpoint table from the database. Use the `DBLOGIN` command to establish a database connection before using this command.

If the checkpoint table is deleted while Replicat is still running and transactions are occurring, Replicat will abend with an error that the checkpoint table could not be found. However, the checkpoints are still maintained on disk in the checkpoint file. To resume processing, add the checkpoint table back under the same name. Data in the trail resumes replicating. Then, you can delete the checkpoint table.

Syntax

```
DELETE CHECKPOINTTABLE [[container. | catalog.]owner.table] [!]
```

container. | catalog.
The Oracle pluggable database, if applicable. If this option is omitted, the catalog or pluggable database defaults to the one that is associated with the `SOURCEDB`, `USERID`, or `USERIDALIAS` portion (depending on the database) of the `DBLOGIN` command.

owner.table
The owner and name of the checkpoint table to be deleted. An owner and name are not required if they are the same as those specified with the `CHECKPOINTTABLE` parameter in the `GLOBALS` file.

!
Bypasses the prompt that confirms intent to delete the table.

Example

```
DELETE CHECKPOINTTABLE ggadmin.ggs_checkpoint
```

DELETE CREDENTIALS

To remove a credential from the local wallet, use the `DELETE CREDENTIALS` command.

Syntax:

```
DELETE CREDENTIALS credential-name
```

credential-name is the name of the credential being deleted.

Example:

```
OGG (not connected) 1> DELETE CREDENTIALS admin
2019-02-14T00:31:36Z  INFO    OGG-15114  Credential store altered.

OGG (not connected) 2>
```

In this example, the `admin` user credential is deleted.

DELETE CREDENTIALSTORE

Use the `DELETE CREDENTIALSTORE` command to remove a credential store from the system. The credential store wallet and its contents are permanently deleted.

The use of a credential store is not supported for the NonStop platforms.

Syntax

```
DELETE CREDENTIALSTORE
```

DELETE DISTPATH

Use `DELETE DISTPATH` to remove a distribution path.

Syntax

```
DELETE DISTPATH path-name
```

The name of the distribution path.

Example

```
DELETE DISTPATH dpe
```

DELETE ENCRYPTIONPROFILE

Use `DELETE ENCRYPTIONPROFILE` to remove an encryption profile.

Syntax

```
DELETE ENCRYPTIONPROFILE encryption-profile-name
```

DELETE EXTRACT

Use `ADD EXTRACT` to create a source database capture process, known as an Extract. The Extract can be created as a change data Extract, which captures current transactions from the database log or through other means depending on the database vendor, or it can be created as an initial load Extract, which will capture the records that exist in the database tables.

It is recommended to create only one change data or one initial load Extract per source database, however in rare situations, it may improve capture throughput by using multiple Extracts per database.

Oracle GoldenGate can support a large number of concurrent Extract and Replicat processes per deployment, depending on the resources available with the operating system. However, it is recommended to monitor system resources as more Extract or Replicat processes get added, and to keep the total number of processes per deployment to 300 or less.

> **Note**
>
> - This command cannot exceed 500 bytes in size for all keywords and input, including any text that you enter for the `DESC` option.
> - For Db2 for i, this command establishes a global start point for all journals and is a required first step. After issuing the `ADD EXTRACT` command, you can then optionally position any given journal at a specific journal sequence number by using the `ALTER EXTRACT` command with an appropriate journal option.
> - For Oracle and PostgreSQL databases, establish a connection to the source database using DBLOGIN USERIDALIAS and then issue the `REGISTER EXTRACT` command before adding the Extract. For details, see REGISTER EXTRACT

Syntax

```
DELETE EXTRACT group_name [!]
```

group_name
The name of an Extract group or a wildcard specification (*) to specify multiple groups. For example, `T*` deletes all Extract groups whose names start with T.

!
(Exclamation point) Deletes all Extract groups associated with a wildcard without prompting.

DELETE EXTTRAIL

Use `DELETE EXTTRAIL` to delete the record of checkpoints associated with a trail on a local system. Checkpoints are maintained in a file bearing the same name as the group in the `dirchk` sub-directory of the Oracle GoldenGate directory.

This command only deletes references to the specified trail from the checkpoint file. It does not delete the trail files.

Syntax

```
DELETE EXTTRAIL trail_name
```

trail_name
The relative or fully qualified path name of the trail, including the two-character trail prefix.

Example

```
DELETE EXTTRAIL north/ea
```

DELETE HEARTBEATENTRY

Valid for all supported databases. See the Certification Matrix for details on supported databases.

Use DELETE HEARTBEATENTRY to delete the records in the heartbeat table with the specified process name either in the incoming or outgoing path columns. This command required a DBLOGIN. On a CDB database, a PDB login is required.

Oracle GoldenGate for Oracle AI Database simplifies the administration of the heartbeat table by eliminating the need for GGSCHEMA or HEARTBEATTABLE parameter. To implement this, Extracts and Replicat look in the schema of the ER processes connected user for the heartbeat tables, except for Oracle CDB root Extract. In case of CDB root Extract, GGSCHEMA is used.

Syntax

```
DELETE HEARTBEATENTRY group_name
```

group_name
The name of the process to be cleaned.

!
(Exclamation point) Deletes all heartbeat table entries associated with a wildcard without prompting.

DELETE HEARTBEATTABLE

Valid for all supported databases. See the Certification Matrix for details on supported databases.

Use DELETE HEARTBEATTABLE to delete tables, procedures, schedulers, and views. This command requires a DBLOGIN. On a CDB database, a PDB login is required.

Oracle GoldenGate for Oracle AI Database simplifies the administration of the heartbeat table by eliminating the need for GGSCHEMA or HEARTBEATTABLE parameter. To implement this, Extracts and Replicat look in the schema of the ER processes connected user for the heartbeat tables, except for Oracle CDB root Extract. In case of CDB root Extract, GGSCHEMA is used.

Syntax

```
DELETE HEARTBEATTABLE heartbeattable_name
```

group_name
The name of the process to be cleaned.

!
(Exclamation point) Deletes all heartbeat table entries associated with a wildcard without prompting.

DELETE MASTERKEY

Use the DELETE MASTERKEY command to mark a version of a master key for deletion. Routinely deleting older versions of a master key ensures that they cannot be used maliciously.

The OPEN WALLET command must be used before using this command or any of the commands that add or renew the master keys or purge the wallet.

To view the version of a master key, use the INFO MASTERKEY command.

This command marks a version for deletion, but does not physically remove it from the wallet, see PURGE WALLET to remove the master key version permanently.

> **Note**
>
> For Oracle GoldenGate deployments using a shared wallet, the older versions of the master key should be retained after the master key is renewed until all processes are using the newest version. The time to wait depends on the topology, latency, and data load of the deployment. A minimum wait of 24 hours is a conservative estimate, but you may need to perform testing to determine how long it takes for all processes to start using a new key. To determine whether all of the processes are using the newest version, view the report file of each Extract immediately after renewing the master key to confirm the last SCN that was mined with the old key. Then, monitor the Replicat report files to verify that this SCN was applied by all Replicat groups. At this point, you can delete the older versions of the master key.

See UNDELETE MASTERKEY to reverse a deletion made by `DELETE MASTERKEY`.

Once a version number is used, the wallet reserves it forever, and no other key of the same version can be generated. For example, you cannot mark version 2 of a key for deletion, then purge the wallet to remove it, and then issue `RENEW MASTERKEY` to add a version 2 again. Even though only version 1 of the key remains in the wallet after the purge, the renewal generates version 3, not version 2.

The use of a wallet and master key is not supported for the NonStop platforms.

Syntax

```
DELETE MASTERKEY
{VERSION version | RANGE FROM begin_value TO end_value | ALL}
```

`VERSION` *`version`*
Specifies a single version to be marked for deletion.

`RANGE FROM` *`begin_value`* `TO` *`end_value`*
Specifies a range of versions to be marked for deletion. The versions must be contiguous. For example, specifying `RANGE FROM 3 TO 6` marks versions 3, 4, 5, and 6.

`ALL`
Marks all versions of the master key for deletion, including the currently active one. When this option is used, it should always be followed by a `RENEW MASTERKEY` command to create a new, current version of the master key.

Examples

This command marks one version of the master key for deletion and returns a message similar to the one shown.

```
DELETE MASTERKEY VERSION 10
Version 10 of Masterkey 'OGG_DEFAULT_MASTERKEY' deleted from wallet at
location './dirwlt'.
```

This command marks versions 3, 4, 5, and 6 for deletion and returns a message similar to the one shown.

```
DELETE MASTERKEY RANGE FROM 3 TO 6
```

Example: Admin Client

```
DELETE MASTERKEY ALL
2019-11-21T19:38:08Z  INFO OGG-06148  Version 1 of master key
'OGG_DEFAULT_MASTERKEY' in Oracle Wallet was deleted.
```

DELETE PROCEDURETRANDATA

Valid for Oracle.

Use `DELETE PROCEDURETRANDATA` to remove supplemental logging for Procedural Replication.

Use the `DBLOGIN` command to establish a database connection before using this command.

Syntax

```
DELETE PROCEDURETRANDATA
```

DELETE PROFILE

This command removes a managed process profile.

Syntax

```
DELETE PROFILE profile-name
```

`profile-name` is the name of the profile being removed.

Example

```
OGG (https://localhost Local) 7> DELETE PROFILE Critical
```

DELETE RECVPATH

Use `DELETE RECVPATH` to remove target-initiated distribution path in the Distribution Service.

Syntax

```
DELETE RECVPATH path-name
```

The name of the distribution path.

Example

```
DELETE RECVPATH rpe
```

DELETE REPLICAT

Use `DELETE REPLICAT` to delete a Replicat group. This command deletes the checkpoint file but leaves the parameter file intact. Then you can re-create the group or delete the parameter file as needed. This command frees up trail files for purging, because the checkpoints used by the deleted group are removed (assuming no other processes are reading the file).

Use the `DBLOGIN` command before deleting any Replicats so that the checkpoint data or any internal information stored in the database for that Replicat can also be cleaned up.

Before using `DELETE REPLICAT`, stop Replicat with the `STOP REPLICAT` command.

If this is an integrated Replicat (Oracle only) or a non-integrated Replicat that uses a checkpoint table, do the following after you stop Replicat:

1. Log into the database by using the `DBLOGIN` command. `DBLOGIN` enables `DELETE REPLICAT` to delete the checkpoints from the checkpoint table of a non-integrated Replicat or to delete the inbound server that an integrated Replicat uses.
2. Issue `DELETE REPLICAT`.

Syntax

```
DELETE REPLICAT group_name [!]
```

group_name
The name of a Replicat group or a wildcard (*) to specify multiple groups. For example, `T*` deletes all Replicat groups whose names begin with T.

!
Use this option to force the Replicat group to be deleted if the `DBLOGIN` command is not issued before the `DELETE REPLICAT` command is issued. If the group is a non-integrated Replicat, this option deletes the group's checkpoints from the checkpoint file on disk, but not from the checkpoint table in the database. If using this option to delete an integrated Replicat group, you must use the `UNREGISTER REPLICAT` command to delete the inbound server from the target database. This option can also be used to ignore the prompt that occurs when a wildcard specifies multiple groups.

> **Note**
>
> The basic `DELETE REPLICAT` command commits an existing Replicat transaction, but the `!` option prevents the commit.

Example

```
DELETE REPLICAT repe
```

DELETE RMTTRAIL

Use DELETE RMTTRAIL to delete the record of checkpoints associated with a trail on a remote system. Checkpoints are maintained in a file bearing the same name as the group in a sub-directory of the Oracle GoldenGate directory.

This command only deletes references to the specified trail from the checkpoint file. It does not delete the trail files.

Syntax

```
DELETE RMTTRAIL trail_name[,EXTRACT group_name}
```

trail_name
The relative or fully qualified path name of the trail, including the two-character trail prefix.

group_name
The name of the Extract group to which the trail is bound. If not specified, DELETE RMTTRAIL deletes the trail reference from all Extract groups that write to the specified trail.

Example

```
DELETE RMTTRAIL var/lib/data/et


DELETE RMTTRAIL north/ea
```

The following command is run from the Admin Client.

```
DELETE RMTTRAIL ea
2019-11-20T23:49:41Z  INFO    OGG-08100 Deleting extract trail ea for Extract
exte
```

DELETE SCHEMATRANDATA

Valid for Oracle.

Use DELETE SCHEMATRANDATA to remove the Oracle schema-level supplemental logging that was added with the ADD SCHEMATRANDATA command. Use the DBLOGIN command to establish a database connection before using this command. The user that is specified with this command must have the privilege to remove supplemental log groups.

By default, this command attempts to remove the supplemental logging of the key columns that are used by Oracle GoldenGate (can be the primary key, a unique key, KEYCOLS columns, or all columns) and also the scheduling columns. The scheduling columns are the primary key, all of the unique keys, and all of the foreign keys. To delete the logging of the Oracle GoldenGate key columns, but not the scheduling columns, include the NOSCHEDULINGCOLS option with DELETE SCHEMATRANDATA. If ADD SCHEMATRANDATA was issued with the ALLCOLS option, use DELETE SCHEMATRANDATA with the ALLCOLS option to remove the supplemental logging of all of the columns, including the Oracle GoldenGate key columns.

Syntax

```
DELETE SCHEMATRANDATA schema [NOSCHEDULINGCOLS | ALLCOLS]
```

schema
The schema for which you want supplemental logging to be removed. Do not use a wildcard. If the source is an Oracle multitenant container database, make certain to log into the pluggable database that contains the schema for which you want to remove the logging.

NOSCHEDULINGCOLS
Prevents the command from removing the supplemental logging of the scheduling columns of the tables in the specified schema. The scheduling columns are the primary key, all of the unique keys, and all of the foreign keys of a table.

ALLCOLS
Removes the supplemental logging of all of the columns of the tables in the specified schema.

Examples

```
DELETE SCHEMATRANDATA hr
```

```
DELETE SCHEMATRANDATA hr ALLCOLS
```

DELETE TRANDATA

Use `DELETE TRANDATA` to do one of the following:

- Db2 LUW and Db2 z/OS: Alters the table to `DATA CAPTURE NONE`.
- Oracle: Disable supplemental logging.
- SQL Server: Stops extended logging for a table.
- PostgreSQL: Alters the table's `REPLICA IDENTITY` to `NOTHING`.

By default, this command attempts to remove the supplemental logging of the key columns that are used by Oracle GoldenGate (can be the primary key, a unique key, `KEYCOLS` columns, or all columns) and also the scheduling columns. The scheduling columns are the primary key, all of the unique keys, and all of the foreign keys. To delete the logging of the Oracle GoldenGate key columns, but not the scheduling columns, include the `NOSCHEDULINGCOLS` option with `DELETE TRANDATA`. If `ADD TRANDATA` was issued with the `ALLCOLS` option, use `DELETE TRANDATA` with the `ALLCOLS` option to remove the supplemental logging of all of the columns, including the Oracle GoldenGate key columns. `DELETE TRANDATA` will disable logical replication for a table after all the table level supplemental logging has been disabled or removed. This behavior is supported from Oracle 19c and higher. Also, if a `DELETE TRANDATA` operation removes the last supplemental log group on a table then it will also perform an `ALTER TABLE` *`owner.table`* `DISABLE LOGICAL REPLICATION` too.

It is mandatory to run `ADD TRANDATA` *`table_name`* command to enable logical replication after running `DELETE TRANDATA`.

> **Note**
>
> You cannot enable logical replication using `ADD SCHEMATRANDATA`.

Use the `DBLOGIN` command to establish a database connection before using this command. The user specified with this command must have the same privileges that are required for `ADD TRANDATA`.

Syntax

```
DELETE TRANDATA [container.]owner.table [NOSCHEDULINGCOLS | ALLCOLS]
```

[container.]owner.table
The pluggable database (if this is an Oracle multitenant container database), owner and name of the table or file. A wildcard can be used for any name component.

`NOSCHEDULINGCOLS`
Prevents the command from removing the supplemental logging of the scheduling columns of the specified table. The scheduling columns are the primary key, all of the unique keys, and all of the foreign keys of a table.

`ALLCOLS`
Removes the supplemental logging of all of the columns of the specified table.

Examples

```
DELETE TRANDATA hr.employees
```

```
DELETE TRANDATA hr.reg*
```

```
DELETE TRANDATA hr.jobs ALLCOLS
```

DISABLE SERVICE

Use `DISABLE SERVICE` to disable the specified Oracle GoldenGate services for any update operations by users without the Security role. You must have the Security role to use this command. Make sure that you stop the service before it can be disabled.

Syntax

```
DISABLE SERVICE service_name_wildcard
```

`service_name_wildcare`

The name of a service or a wildcard (*) to specify multiple services. Valid services are `ADMINSRVR`, `DISTSRVR`, `RECVSRVR`, and `PMSRVR`.

Example

```
DISABLE SERVICE ADMIN*
```

DISCONNECT

Use to disconnect from the Service Manager. It is not necessary to disconnect from one Service Manager connection to connect to another Service Manager. Use the `CONNECT` command to establish a connection to a Service Manager.

Syntax

```
DISCONNECT
```

EDIT ENCKEYS

Use `EDIT ENCKEYS` to open the `ENCKEYS` file for editing in the default text editor.

Syntax

```
EDIT ENCKEYS
```

EDIT GLOBALS

Use this to open the `GLOBALS` parameter file for editing in the default text editor. The default text editor is set using the `SET EDITOR` command.

After the globals file has been modified, when you leave the editor, the globals parameter file is checked to ensure its valid.

Syntax

```
EDIT GLOBALS
```

Examples

The following examples show how Admin Client checks the validity of the parameter file.

When you run the `EDIT GLOBALS` command and the paramater file is valid, then the editor opens up to allow editing the GLOBALS parameters. If the file doesn't pass the validity check, then the following error is displayed:

The output displays:

```
2019-11-21T21:01:21Z  ERROR   OGG-10143  (GLOBALS) line 2: Parameter
[junkasdsad] is unrecognized.  No parameter definition with that name could
be found.

2019-11-21T21:01:21Z  ERROR   OGG-10184  Parameter file GLOBALS:  Validity
check: FAIL
```

EDIT PARAMS

Use `EDIT PARAMS` to create or change a parameter file. By default, the editor is set with your `EDITOR` environment variable. You can change the default editor with the `SET EDITOR` command.

Syntax

```
EDIT PARAMS group_name
```

group_name
Opens a parameter file for the specified Extract or Replicat group. Use this command option to create or edit an existing parameter file that is in a character set of the local operating system.

Examples

```
EDIT PARAMS exte
```

ENABLE SERVICE

Use `ENABLE SERVICE` to enable the specified Oracle GoldenGate services. You must have the Security role to use this command.

Syntax

```
ENABLE SERVICE service_name_wildcard
```

service_name_wildcard

The name of a service or a wildcard (*) to specify multiple services. Valid services are `ADMINSRVR`, `DISTSRVR`, `RECVSRVR`, and `PMSRVR`.

Example

```
START SERVICE ADMIN*
```

ENCRYPT PASSWORD

Use `ENCRYPT PASSWORD` to encrypt a password that is used in an Oracle GoldenGate parameter file or command.

Admin Client Syntax

```
ENCRYPT PASSWORD password
ENCRYPTKEY key_name
```

password
The login password. Do *not* enclose the password within quotes. Do *not* use commas in passwords. If the password is case-sensitive, type it that way.

ENCRYPTKEY *{key_name}*
Specifies the encryption key.

key_name
Specifies the logical name of a user-created encryption key in a local `ENCKEYS` lookup file. The key name is used to look up the actual key in the `ENCKEYS` file. A user-created key and an associated `ENCKEYS` file is required when using AES encryption. To use *`key_name`*, generate the key with `KEYGEN` or another utility, then store it in an `ENCKEYS` file on the source and target systems. The AES ciphers have a 128-bit, 192-bit, and 256-bit block size.

Examples

```
ENCRYPT PASSWORD ny14072 ENCRYPTKEY superkey2
```

EXIT

Use `EXIT` to exit the Oracle GoldenGate command line interface.

Syntax

```
EXIT
```

FLUSH SEQUENCE

This command updates an Oracle sequence so that initial redo records are available at the time the Extract starts to capture transaction data. Normally, redo is not generated until the current cache is exhausted. The flush gives Replicat an initial start point with which to synchronize to the correct sequence value on the target system. From then on, Extract can use the redo that is associated with the usual cache reservation of sequence values.

Syntax

```
FLUSH SEQUENCE owner.sequence
```

owner.sequence
The owner and name of a sequence. The schema name cannot be null and is required You can use an asterisk (*) wildcard for the sequence name, but not for the owner name.

Example

```
FLUSH SEQUENCE ggadmin.seq
```

HELP

Use `HELP` to obtain information about an Oracle GoldenGate command. The basic command returns a list of commands. The *command* option restricts the output to that of a specific command.

Syntax

```
HELP [command]
```

command
The command that you want help for. You can use a wildcard (*).

Example

To display all commands that begin with `ADD`.

```
HELP ADD
```

HEALTH DEPLOYMENT

Use to display the health of the specified Oracle GoldenGate deployments. Only enabled services will show up in the results, not the disabled services.

Syntax

```
HEALTH DEPLOYMENT deployment-name-wildcard
```

deployment-name-wildcard
The name of the deployment you are interested in. You can use an asterisk (*) wildcard for any portion of the deployment name.

Example

```
HEALTH DEPLOYMENT Phoenix
```

HISTORY

Use `HISTORY` to view a list of the most recently issued commands since the startup of the session. You can use the ! command to re-execute a command in the list.

Syntax

```
HISTORY [n]
```

n
Returns a specific number of recent commands, where *n* is any positive number.

Example

```
HISTORY 7
```

The result of this command would be similar to:

```
1: ADD EXTRACT
2: ADD EXTTRAIL
3: INFO CHECKPOINTTABLE
4: EDIT PARAMS
5: START EXTRACT
6: HISTORY
```

INFO ALL

Use `INFO ALL` to display the status and lag (where relevant) for Extract, Replicat, DISTPATH, and RECVPATH processes on a system. When Oracle Grid Infrastructure Agents (XAG) Clusterware components are in use, the relevant information is also displayed.

The basic command, without options, displays only online (continuous) processes and Microservices. To display tasks, use either `INFO ALL TASKS` or `INFO ALL ALLPROCESSES`.

The `Status` and `Lag at Chkpt` (checkpoint) fields display the same process status and lag as the `INFO EXTRACT` and `INFO REPLICAT` commands.

If Replicat is in coordinated mode, `INFO ALL` shows only the coordinator thread. To view information about individual threads, use INFO REPLICAT.

Syntax

```
INFO ALL [TASKS | ALLPROCESSES]
```

TASKS
Displays information only for tasks.

ALLPROCESSES
Displays information for online processes and tasks.

Examples

```
INFO ALL TASKS
```

```
INFO ALL ALLPROCESSES
```

INFO AUTHORIZATIONPROFILE

Lists all the authorization profiles in a deployment or information on a specific authorization profile for a specific deployment.

Syntax:

```
INFO AUTHORIZATIONPROFILE ( profile-name | ALL )
DEPLOYMENT deployment-name
```

profile-name
Name of the authorization profile for which the details need to be displayed.

ALL
Lists all existing profiles in the deployment. It does not get detailed information about them nor list the currently enabled profile.

deployment-name
Name of the deployment for which the information is needed.

Examples

The following examples lists out all authorization profiles in Service Manager.

```
INFO AUTHORIZATIONPROFILE ALL DEPLOYMENT ServiceManager
```

The following example gets the information about the testProfile in Service Manager. In this deployment, there are two profiles: `localCredentialStore` and `testProfile`.

```
INFO AUTHORIZATIONPROFILE testProfile DEPLOYMENT ServiceManager
```

The output for this command is:

```
Type:                idcs
      Enabled:               true
     Client ID:              1234567890abcdefghijklmnopqrstuv
     Tenant Discovery URI:   https://prodserver.idcs.com/.well-known/openid-
configuration

     Group To Role Mapping:
       Security:            Security-Group
```

INFO CHECKPOINTTABLE

Not valid for Replicat for Java, Oracle GoldenGate Applications Adapter, or Oracle GoldenGate Big Data.

Use `INFO CHECKPOINTTABLE` to confirm the existence of a checkpoint table and view the date and time that it was created. It returns a message similar to the following:

```
Checkpoint table ggadmin.ggs_checkpoint created 2017-01-06T11:51:53.
```

Use the `DBLOGIN` command to establish a database connection before using this command.

Syntax

```
INFO CHECKPOINTTABLE [[container. | catalog.]owner.table]
```

container. | catalog.
The Oracle pluggable database, if applicable. If this option is omitted, the catalog or pluggable database defaults to the one that is associated with the SOURCEDB, USERID, or USERIDALIAS portion of the DBLOGIN command (depending on the database).

owner.table
The owner and name of the checkpoint table. An owner and name are not required if they are the same as those specified with the CHECKPOINTTABLE parameter in the GLOBALS file. You can use a wildcard (*).

Example

```
INFO CHECKPOINTTABLE ggadmin.ggs_checkpoint
```

INFO CREDENTIALS

The INFO CREDENTIALS command retrieves a list of Oracle GoldenGate user credentials.

Example

```
OGG (not connected) 1> INFO CREDENTIALS *
```

Output:

```
Credentials Name     Type             User Name
admin                username         oggadmin
admin                token
```

When mixed wildcard is used:

```
OGG (not connected) 2> INFO CREDENTIALS a*
```

Output:

```
Credentials Name         User Name
admin                            oggadmin
```

INFO CREDENTIALSTORE

Use the INFO CREDENTIALSTORE command to get information about an Oracle GoldenGate credential store. This information includes the aliases that a credential store contains and the user IDs that correspond to them. The encrypted passwords in the credential store are not returned.

The credential store location is identified by the CREDENTIALSTORELOCATION parameter in the GLOBALS file.

The use of a credential store is not supported for the NonStop platforms.

Syntax

```
INFO CREDENTIALSTORE [DOMAIN domain]
```

DOMAIN *domain*
Returns the aliases and user IDs for a specific domain. For security purposes, if the DOMAIN option is omitted, only the aliases and user IDs under the default domain of OracleGoldenGate are shown. It is not possible to see DOMAIN credentials unless the person issuing the INFO CREDENTIALSTORE command knows the name of the domain. See ALTER CREDENTIALSTORE for more information about domains.

Examples

```
INFO CREDENTIALSTORE
```

```
INFO CREDENTIALSTORE DOMAIN support
```

Example: Command run in the Admin Client
The following example shows credential store domain as Oracle GoldenGate with the alias set up as ggeast and the user ID as ggadmin.

```
INFO CREDENTIALSTORE DOMAIN OracleGoldenGate
Default domain: OracleGoldenGate
  Alias: ggeast
  Userid: ggadmin
```

INFO DISTPATH

Use INFO DISTPATH to return information about distribution paths. Before you run this command, ensure that the Distribution Server is running for that deployment.

Syntax

```
INFO DISTPATH ALL | DISTPATH-NAME [DETAIL]
```

ALL
Displays a list of all distribution paths with their status.

DISTPATH-NAME
The name of the distribution path.

DETAIL
(Optional) Displays the following additional information for the requested distribution path:

- Process and thread information.
- Source database name where the data is originated.
- Last started timestamp and processing lag.
- Current and starting input and output checkpoint.

INFO ENCRYPTIONPROFILE

Use `INFO ENCRYPTIONPROFILE` to return information about the encryption profile.

Syntax

```
INFO ENCRYPTIONPROFILE encryption-profile-name-wildcard [DETAIL ]
```

`encryption-proifle-name-wildcarded`
Displays a list of all encryption profiles.

INFO ER

Use the `INFO ER` command to get information on multiple Extract and Replicat groups as a unit. Use it with wildcards to affect every Extract and Replicat group that satisfies the wildcard. This information is returned:

- The status of Extract (`STARTING`, `RUNNING`, `STOPPED` or `ABENDED`). `STARTING` means that the process has started but has not yet locked the checkpoint file for processing.
- Approximate Extract lag.
- Checkpoint information.
- Process run history.
- The trails to which Extract is writing.
- Status of upgrade to, or downgrade from, integrated capture. The process can be running or stopped when `INFO ER` is issued. With a running process, the status of `RUNNING` can mean one of the following:
- Active: Running and processing (or able to process) data. This is the normal state of a process after it is started.
- Suspended: The process is running, but suspended due to an `EVENTACTIONS SUSPEND` action. In a suspended state, the process is not active, and no data can be processed, but the state of the current run is preserved and can be continued by issuing the `SEND` command with the `RESUME` option. The RBA in the `INFO` command reflects the last checkpointed position before the suspend action. To determine whether the state is active or suspended, issue the `SEND` command with the `STATUS` option.

Syntax

```
INFO ER group_wildcard
[, SHOWCH checkpoints-number]
[, DETAIL]
[, TASKS]
[, ALLPROCESSES]
```

group_wildcard
The wildcard specification for the groups that you want to affect with the command. Oracle GoldenGate automatically increases internal storage to track up to 100,000 wildcard entries.

SHOWCH *checkpoint-number*
The basic command shows information about the current Extract checkpoints. Extract checkpoint positions are composed of read checkpoints in the data source and write checkpoints in the trail. The trail type, `EXTTRAIL` is also noted.
Optionally, specify a value for checkpoint number to include the specified number of previous checkpoints as well as the current one.

> **Note**
>
> You might see irregular indents and spacing in the output. This is normal and does not affect the accuracy of the information.

DETAIL
Displays the Extract run history, including start and stop points in the data source expressed as a time and the trails to which Extract is writing.

TASKS
Displays Extract tasks only. Tasks that were specified by a wildcard argument are not displayed by `INFO EXTRACT`.

ALLPROCESSES
Displays all Extract groups, including tasks.

Example

```
INFO ER *
```

Output would be similar to the following:

```
Extract    EXTE      Last Started 2022-06-07 07:33   Status RUNNING
Description          east
Checkpoint Lag       00:00:00 (updated 00:00:05 ago)
Process ID           60782
Log Read Checkpoint  Oracle Integrated Redo Logs
                     2022-09-05 23:17:19
                     SCN 0.89996333 (89996333)
Settings Profile     Default
Encryption Profile   LocalWallet


Replicat   REPE      Last Started 2022-06-15 03:45   Status RUNNING
Description          east
INTEGRATED
Parallel
Checkpoint Lag       00:00:00 (updated 00:00:08 ago)
Process ID           96740
Log Read Checkpoint  File east/ea000000009
                     2022-09-05 23:17:19.808198  RBA 9382
Settings Profile     Default
```

```
Encryption Profile   LocalWallet
```

INFO EXTRACT

Use `INFO EXTRACT` to view the following information.

- The status of Extract (`STARTING`, `RUNNING`, `STOPPED` or `ABENDED`). `STARTING` means that the process has started but has not yet locked the checkpoint file for processing.
- Approximate Extract lag.
- Checkpoint information.
- Process run history.
- The trails to which Extract is writing.
- Status of upgrade to, or downgrade from, Integrated Extract

Syntax

```
INFO EXTRACT group_name
[, SHOWCH [checkpoint_number]]
[, DETAIL]
[, TASKS | ALLPROCESSES]
[, UPGRADE | DOWNGRADE]
[, CONTAINERS]
```

group_name
The name of an Extract group or a wildcard (*) to specify multiple groups. For example, `T*` shows information for all Extract groups whose names start with T. You can list the PDBs registered with a specified Extract group name.

`SHOWCH [`*`checkpoint_number`*`]`
The basic command shows information about the current Extract checkpoints. Extract checkpoint positions are composed of read checkpoints in the data source and write checkpoints in the trail. The trail type `EXTTRAIL` is also noted.
Optionally, specify a value for *`checkpoint_number`* to include the specified number of previous checkpoints as well as the current one.

> **Note**
>
> You might see irregular indents and spacing in the output. This is normal and does not affect the accuracy of the information.

See About Checkpoints, which includes descriptions of the types of checkpoints made by each process and the internal metadata entries that are included in the display.

`DETAIL`
Displays the following:

- Extract run history, including start and stop points in the data source, expressed as a time.
- Trails to which Extract is writing.

TASKS
Displays only Extract tasks. Tasks that were specified by a wildcard argument are not displayed by `INFO EXTRACT`.

ALLPROCESSES
Displays all Extract groups, including tasks.

UPGRADE | DOWNGRADE
Valid for an Oracle Database only.

- `UPGRADE` displays whether the Extract can be upgraded from classic capture mode to integrated capture mode.
- `DOWNGRADE` displays whether the Extract can be downgraded from integrated capture mode to classic capture mode.

If Extract cannot be upgraded or downgraded, the reason why is displayed.
A wildcarded Extract name is not allowed with this option.
Before using this command, issue the `DBLOGIN` command.

CONTAINERS
Lists the PDBs that are registered with the specified Extract group. However, the command errors if it is run in non-CDB mode or the Extract group doesn't exist. Issue the `DBLOGIN` command before using this option.

Extract can be running or stopped when `INFO EXTRACT` is issued. In the case of a running process, the status of `RUNNING` can mean one of the following:

- Active: Running and processing (or able to process) data. This is the normal state of a process after it is started.
- Suspended: The process is running, but suspended due to an `EVENTACTIONS SUSPEND` action. In a suspended state, the process is not active, and no data can be processed, but the state of the current run is preserved and can be continued by issuing the `SEND EXTRACT` command with the `RESUME` option. The RBA in the `INFO` command reflects the last checkpointed position before the suspend action. To determine whether the state is active or suspended, issue the `SEND EXTRACT` command with the `STATUS` option.

The basic command displays information only for online (continuous) Extract processes. Tasks are excluded.

> **Note**
>
> When a connection string includes multiple hosts, but some hosts are unreachable while some are accessible, the `INFO EXTRACT` command fails to retrieve the process status. To check the process status in such cases, you must create a separate *alias* that includes only the reachable host.

About Extract Lag
The `Checkpoint Lag` field of the `INFO EXTRACT` output reflects the lag, in seconds, at the time that the last checkpoint was written to the trail. For example, if the following is true:

- Current time = 15:00:00
- Last checkpoint = 14:59:00
- Timestamp of the last record processed = 14:58:00

Then the lag is reported as 00:01:00 (one minute, the difference between 14:58 and 14:59).

A lag value of UNKNOWN indicates that the process could be running but has not yet processed records, or that the source system's clock is ahead of the target system's clock (due to clock imperfections, not time zone differences), see LAG EXTRACT. For a more precise calculation of the lag and for determining the overall lag, use the heartbeat table. See ADD HEARTBEATTABLE.

Examples

```
INFO EXTRACT ext*, SHOWCH
```

```
INFO EXTRACT *, TASKS
```

```
INFO EXTRACT exte UPGRADE
```

The following example shows the use of the INFO EXTRACT command from the Admin Client.

```
INFO EXTRACT exte

EXTRACT    exte  Initialized   2019-11-20 23:22   Status STOPPED
Checkpoint Lag       00:00:00 (updated 00:00:09 ago)
Log Read Checkpoint  Oracle Redo Logs
                     2019-11-20 23:22:54  Seqno 0, RBA 0
                     SCN 0.0 (0)
Auto Start           Delay:   0
Encryption Profile   LocalWallet
```

The following example shows the command output with DBLOGIN for PostgreSQL:

```
EXTRACT extw
Last Started 2020-07-01 01:40 Status RUNNING
Checkpoint Lag 00:00:00 (updated 00:00:09 ago)
Process ID 101094
VAM Read Checkpoint 2020-07-01 01:40:15.170802

Replication Slot edf_tc1_2c59ae112747afa1 is active with PID 101120 in
database postgres
Slot Restart LSN 1/B982ED98
Slot Flush LSN 1/B982ED98
Current Log Position 1/B982EDD0
```

The following example shows the command output without DBLOGIN for PostgreSQL:

```
EXTRACT extw Last Started 2020-07-01 01:40 Status RUNNING
Checkpoint Lag 00:00:00 (updated 00:00:04 ago)
Process ID 101094
VAM Read Checkpoint 2020-07-01 01:44:16.045693
LSN: 1/B99B7E60
Replication Slot Unavailable(requires DBLOGIN)
```

```
Slot Restart LSN Unavailable(requires DBLOGIN)
Slot Flush LSN Unavailable(requires DBLOGIN)
Current Log Position Unavailable(requires DBLOGIN)
```

INFO EXTTRAIL

Use `INFO EXTTRAIL` to retrieve configuration information for a local trail. It shows the name of the trail, the Extract that writes to it, the position of the last data processed, and the assigned maximum file size.

Syntax

```
INFO EXTTRAIL trail_name
```

trail_name
The relative or fully qualified path name of the trail or a wildcard designating multiple trails. For example, `T*` shows information for all trail files whose names start with T.

Examples

```
INFO EXTTRAIL north\ea
```

```
INFO EXTTRAIL *
```

The following example shows the use of the `INFO EXTTRAIL` command from the Admin Client where the trail name is `aa`.

```
INFO EXTTRAIL ea
         Local Trail: EA
        Seqno Length: 6
   Flip Seqno Length: yes
             Extract: exte
               Seqno: 0
                 RBA: 0
           File Size: 1500M
```

INFO HEARTBEATTABLE

Valid for all supported databases. See the Certification Matrix for details on supported databases.

Use `INFO HEARTBEATTABLE` to display information about the heartbeat tables configured in the database.

This command requires a `DBLOGIN`. For an Oracle multitenant database, the `DBLOGIN` to a PDB is required.

Oracle GoldenGate for Oracle AI Database simplifies the administration of the heartbeat table by eliminating the need for GGSCHEMA or HEARTBEATTABLE parameter. To implement this, Extracts and Replicat look in the schema of the ER processes connected user for the

heartbeat tables, except for Oracle CDB root Extract. In case of CDB root Extract, GGSCHEMA is used. In case of Autonomous AI Database, the user must be `GGADMIN`.

Syntax

```
INFO HEARTBEATTABLE
```

INFO MASTERKEY

Use the `INFO MASTERKEY` command to view the contents of a currently open master-key wallet. If a wallet store does not exist, a new Wallet store file is created. This Wallet store file is then used to host different encrypted keys as they are created.

The default output shows the version history of the master key, with the creation date of a version and the status of the version. The status can be one of the following:

- Current: Indicates this is the current version of the master key, which is the latest version of the master key as well.

  > **Note**
  >
  > Changing prior key versions to Current status is not allowed.

- Available: Indicates this version is not the current one but can be made active, if needed.
- Deleted: Indicates that this version is marked to be deleted when the PURGE WALLET command is issued.

The use of a wallet and master key is not supported for the NonStop platforms.

Syntax

```
INFO MASTERKEY [VERSION version]
```

`VERSION` *`version`*
Shows detailed information about a specific version of the master key. The output includes the original creation date, the latest renewal date, the status, and the hash of AES (Advanced Encryption Standard) Key.

Examples

```
INFO MASTERKEY
```

```
INFO MASTERKEY VERSION 7
```

Example: Admin Client

```
INFO MASTERKEY
Masterkey Name: OGG_DEFAULT_MASTERKEY
```

```
Version         Creation Date                    Status
1               2019-11-21T19:37:23.000+00:00    Current
```

INFO PARAM

Use `INFO PARAM` to retrieve the parameter definition information. If a name matches multiple records, they are all displayed. If the query parameter has child options, they are not displayed in the output though their names are listed in the Options tab. To display the full record of an option, the full name in the form of *`parameter.option`* should be queried separately.

This parameter infrastructure allows unlimited levels of options. So, the full name of a parameter or option might have numbers of segments, such as A.B.C.D.

Syntax

```
INFO PARAM name
```

name
The name of a parameter, an option, or a full name that is part of the several names concatenated together using dot ('.') as the delimiter.

Example 1

The following example uses `GETINSERTS` with `INFO PARAM` along with the output.

```
INFO PARAM GETINSERTS
```

Output:

```
param name  : getinserts
opposite    : ignoreinserts
description : Include insert records.
argument    : boolean
default     : true
component(s): EXTRACT - All
            : REPLICAT - All
platform(s) : All
database(s) : All
status      : current
mandatory   : false
dynamic     : false
relations   : none
```

Example 2

The following example uses the `DBOPTIONS.DEFERREFCONST` with `INFO PARAM` to show how to get specific details about a parameter option.

```
INFO PARAM DBOPTIONS.DEFERREFCONST
```

Output:

```
param name  : dboptions.deferrefconst
description : Sets constraints to DEFERRABLE to delay the checking and
enforcement of cascade delete and cascade update referential integrity
constraints by the Oracle target database until the Replicat transaction is
committed.
component(s): REPLICAT - All
platform(s) : All
database(s) : Oracle 11g
            : Oracle 12c
            : Oracle 18c
            : Oracle 19c
            : Oracle 21c
status      : current
mandatory   : false
dynamic     : false
relations   : none
```

INFO PROFILE

This command returns information about managed process profiles.

Syntax:

```
INFO PROFILE profile-name-wildcard
```

Example

```
OGG (https://localhost Local) 4> INFO PROFILE *


                                  Auto    Delay     Auto
Wait     Reset     Disable
Name                             Start   Seconds  Restart   Retries
Seconds   Seconds  on Failure
-------------------------------  -----  --------  -------  --------
--------  --------  ----------
Default                             No               No
```

INFO PROCEDURETRANDATA

Use `INFO PROCEDURETRANDATA` to display information about procedure-level supplemental database logging (Procedural Replication).

Use the `DBLOGIN` command to establish a database connection before using this command.

Syntax

```
INFO PROCEDURETRANDATA
```

INFO REPLICAT

Use `INFO REPLICAT` to retrieve the processing history of a Replicat group. The output of this command includes:

- The status of Replicat (`STARTING`, `RUNNING`, `STOPPED` or `ABENDED`). `STARTING` means that the process has started but has not yet locked the checkpoint file for processing.
- (Oracle Database) The Replicat mode: non-integrated or integrated.
- Whether or not Replicat is in coordinated mode and, if so, how many threads it currently uses.
- Approximate Replicat lag.
- The trail from which Replicat is reading.
- Replicat run history, including checkpoints in the trail.
- Information about the Replicat environment.

The basic command displays information only for online (continuous) Replicat groups. Tasks are excluded.

Replicat can be stopped or running when `INFO REPLICAT` is issued. In the case of a running process, the status of `RUNNING` can mean one of the following:

- `Active`: Running and processing (or able to process) data. This is the normal state of a process after it is started.
- `Suspended`: The process is running, but suspended due to an `EVENTACTIONS SUSPEND` action. In a suspended state, the process is not active, and no data can be processed, but the state of the current run is preserved and can be continued by issuing the `RESUME` command. The RBA in the `INFO` command reflects the last checkpointed position before the suspend action. To determine whether the state is active or suspended, issue the `SEND REPLICAT` command with the `STATUS` option.

About Lag

`Checkpoint Lag` is the lag, in seconds, at the time the last checkpoint was written to the checkpoint file. For example, consider the following:

- Current time = 15:00:00
- Last checkpoint = 14:59:00
- Timestamp of the last record processed =14:58:00

Assuming these values, the lag is reported as 00:01:00 (one minute, the difference between 14:58 and 14:59).

A lag value of `UNKNOWN` indicates that Replicat could be running but has not yet processed records, or that the source system's clock is ahead of the target system's clock (due to clock imperfections, not time zone differences).

To learn more about lag information, see LAG REPLICAT. For complete end to end lag, use the heartbeat table functionality. See ADD HEARTBEATTABLE.

Syntax

```
INFO REPLICAT group_name
[, SHOWCH [checkpoint_number]]
```

```
[, DETAIL]
[, TASKS | ALLPROCESSES]
```

group_name
The name of a Replicat group or a wildcard (*) to specify multiple groups. For example, `T*` shows information for all Replicat groups whose names begin with T.

SHOWCH [*checkpoint_number*]
Displays current checkpoint details, including those recorded to the checkpoint file and those recorded to the checkpoint table, if one is being used. The database checkpoint display includes the table name, the hash key (unique identifier), and the create timestamp. Specify a value for checkpoints-number to include the specified number of previous checkpoints as well as the current one.

DETAIL
Displays detail information. For an Oracle target, `DETAIL` displays the name of the inbound server when Replicat is in integrated mode.
If Replicat is in coordinated mode, `DETAIL` will display only the active threads. For example, if a Replicat named `CR` was created with a maximum of 15 threads, but only threads 7-9 are running, `INFO REPLICAT` *`group_name`* with `DETAIL` will show only the coordinator thread (`CR`), `CR007`, `CR008`, and `CR009`. Checkpoints exist for the other threads, but they are not shown in the command output.
To view `LOGBSN` information with the `DETAIL` output, issue the `DBLOGIN` command before you issue `INFO REPLICAT`. If the command is issued for a specific thread ID of a coordinated Replicat, only the `LOGBSN` for that thread is displayed. Otherwise, the `LOGBSN`s for all threads are displayed.

TASKS
Displays only Replicat tasks. Tasks that were specified by a wildcard argument are not displayed by `INFO REPLICAT`.

ALLPROCESSES
Displays all Replicat groups, including tasks.

Examples

```
INFO REPLICAT *, DETAIL, ALLPROCESSES
```

```
INFO REPLICAT *, TASKS
```

```
INFO REPLICAT repe, SHOWCH
```

The following example run the command from the Admin Client.

```
INFO REPLICAT repe

REPLICAT   REPE  Initialized   2019-11-20 23:54   Status STOPPED
Description          demo replicat
Checkpoint Lag       00:00:00 (updated 00:00:14 ago)
```

```
Log Read Checkpoint  File ea000000000
                     First Record  RBA 0
Settings Profile     default_security
Encryption Profile   LocalWallet
```

INFO RMTTRAIL

Use `INFO RMTTRAIL` to retrieve configuration information for a remote trail. It shows the name of the trail, the Extract that writes to it, the position of the last data processed, and the assigned maximum file size.

Syntax

```
INFO RMTTRAIL trail_name
```

trail_name
The relative or fully qualified path name of the trail or a wildcard (*) designating multiple trails.

Examples

```
INFO RMTTRAIL north\ea
```

```
INFO RMTTRAIL *
```

The following is a sample of `INFO RMTTRAIL` output.

```
Extract Trail: /ogg/var/lib/data/ea
 Seqno Length: 9
 Flip Seqno Length: no
 Extract: exte
 Seqno: 4
 RBA: 78066
 File Size: 500M


Extract Trail: /ogg/dirdat/ea
       Seqno Length: 9
  Flip Seqno Length: no
      Extract: exte
        Seqno: 4
          RBA: 78066
    File Size: 500M
```

INFO RECVPATH

Use `INFO RECVPATH` to return information about a target-initiated distribution path in the Receiver Service. Before you run this command, ensure that the Receiver Service is running.

Syntax

```
INFO RECVPATH ALL | PATH-NAME [DETAIL]
```

ALL
Displays a list of target-initiated paths with their status.

PATH-NAME
Specify the name of the path to view its details.

DETAIL
(Optional) Displays this additional information for the requested receiver path:

- Process and thread information.
- Source database name where the data is originated.
- Last started timestamp and processing lag.
- Current and starting input and output checkpoint.

INFO SCHEMATRANDATA

Use `INFO SCHEMATRANDATA` to determine whether Oracle schema-level supplemental logging is enabled for the specified schema or if any instantiation information is available. Use the `DBLOGIN` command to establish a database connection before using this command.

Syntax

```
INFO SCHEMATRANDATA schema
```

schema
The schema for which you want to confirm supplemental logging. Do not use a wildcard. To get information on the appropriate schema in an Oracle multitenant container database, make certain to log into the correct pluggable database with `DBLOGIN`.

Example

```
INFO SCHEMATRANDATA hr
```

INFO TRANDATA

Valid for all supported databases for the current release.

Use `INFO TRANDATA` to get the following information:

- Db2 LUW and Db2 z/OS: Determine whether `DATA CAPTURE` is enabled or not.
- Oracle: Determine whether supplemental logging is enabled, and to show the names of columns that are being logged supplementary. If all columns are being logged, the notation `ALL` is displayed instead of individual column names. Displays any SCN instantiation information.
- SQL Server: Determine whether or not extended logging is enabled for a table.

- PostgreSQL: Determine whether supplemental logging is enabled and to show the current `REPLICA IDENTITY` setting.

Use the `DBLOGIN` command to establish a database connection before using this command.

Syntax

```
INFO TRANDATA [container.]owner.table
```

[container.]owner.table
The pluggable database (if this is an Oracle multitenant container database), owner and name of the table or file for which you want to view trandata information. The owner is not required if it is the same as the login name that was specified by the `DBLOGIN` command. A wildcard can be used for the table name but not the owner name.

Examples

```
INFO TRANDATA hr.employees
```

```
INFO TRANDATA hr.reg*
```

KILL ER

Use the `KILL ER` command to forcefully terminate multiple Extract and Replicat groups as a unit. Use it with wildcards to affect every Extract and Replicat group that satisfies the wildcard.

Terminating a process leaves the most recent checkpoint in place, and the current transaction is rolled back by the database, guaranteeing that no data is lost when the process is restarted. Use this command only if the process cannot be stopped gracefully with the `STOP REPLICAT` command.

Syntax

```
KILL ER group_name
```

group_name
The name of the group to close. A wildcard can be used for the group name. Oracle GoldenGate automatically increases internal storage to track up to 100,000 wildcard entries.

Example

```
KILL ER extegrp
```

KILL EXTRACT

Use `KILL EXTRACT` to end an Extract process. Use this command only if a process cannot be stopped gracefully with the `STOP EXTRACT` command. The Administration Service profile for managed processes will not attempt to restart an ended Extract process.

Syntax

```
KILL EXTRACT group_name
```

group_name
The name of an Extract group or a wildcard (*) to specify multiple groups. For example, `T*` ends all Extract processes whose group names start with T.

Example

```
KILL EXTRACT exte
```

KILL REPLICAT

Use `KILL REPLICAT` to terminate a Replicat process, which leaves the most recent checkpoint in place and the current transaction is rolled back by the database. This guarantees that no data is lost when the process is restarted.

The managed processes profile in Administration Service will not attempt to restart a terminated Replicat process. Use this command only if Replicat cannot be stopped gracefully with the `STOP REPLICAT` command.

Syntax

```
KILL REPLICAT group_name
```

group_name
The name of a Replicat group or a wildcard (*) to specify multiple groups. For example, `T*` terminates all Replicat processes whose group names begin with T.

Example

```
KILL REPLICAT repe
```

LAG ER

Use the `LAG ER` to get lag information on multiple Extract and Replicat groups as a unit. Use it with wildcards to affect every Extract and Replicat group that satisfies the wildcard. For descriptions and optional parameters for this command, see `LAG EXTRACT`.

Syntax

```
LAG  ER group_name
```

group_name
The name of a group or a wildcard (*) to specify multiple groups. Oracle GoldenGate automatically increases internal storage to track up to 100,000 wildcard entries.

Example

```
LAG  ER exte
```

The output is similar to the following:

```
Sending GETLAG request to Extract group EXTE ...

Last record lag 2 seconds.
At EOF, no more records to process
No Replicat groups found, but some coordinated threads may have been excluded.
```

LAG EXTRACT

Use `LAG EXTRACT` to determine a true lag time between Extract and the data source. `LAG EXTRACT` calculates the lag time more precisely than `INFO EXTRACT` because it communicates with Extract directly, rather than reading a checkpoint position in the trail.

For Extract, lag is the difference, in seconds, between the time that a record was processed by Extract (based on the system clock) and the timestamp of that record in the data source.

If the heartbeat functionality is enabled, you can view the associated lags.

Syntax

```
LAG EXTRACT group_name
```

group_name
The name of an Extract group or a wildcard (*) to specify multiple groups. For example, `T*` determines lag time for all Extract groups whose names start with T.

Examples

```
LAG EXTRACT *
```

```
LAG EXTRACT *ext*
```

The following is sample output for `LAG EXTRACT`.

```
Sending GETLAG request to EXTRACT CAPTPCC...
Last record lag: 2 seconds.
At EOF, no more records to process.
```

LAG REPLICAT

Use `LAG REPLICAT` to determine a true lag time between Replicat and the trail. `LAG REPLICAT` estimates the lag time more precisely than `INFO REPLICAT` because it communicates with Replicat directly rather than reading a checkpoint position.

For Replicat, lag is the difference, in seconds, between the time that the last record was processed by Replicat (based on the system clock) and the timestamp of the record in the trail.

If the heartbeat functionality is enabled, you can view the associated lags. A DBLOGIN is required to view the heartbeat lag.

From 21c onward, the DB_UNIQUE_NAME is displayed if it exists for a remote database, otherwise the DB_NAME value is displayed.

Syntax

```
LAG EXTRACT group_name
```

group_name
The name of a Replicat group or a wildcard (*) to specify multiple groups. For example, T* shows lag for all Replicat groups whose names begin with T.

Examples

```
LAG REPLICAT *
```

```
LAG REPLICAT *repe*
```

LIST TABLES

Use LIST TABLES to list all tables in the database that match the specification provided with the command argument. Use the DBLOGIN command to establish a database connection before using this command. If logging into an Oracle multitenant container database, log in to the pluggable database that contains the tables that you want to list.

If you want to list all the tables that are enabled for auto capture, then use the auto_capture option. This option is only valid for Oracle Database 21c or higher. You need to establish a database connection (using DBLOGIN command) before using this command. If you are unable to establish a connection or don't have the required privileges, the option will not work.

Syntax

```
LIST TABLES table
```

The following syntax applies when using the auto_capture option.

```
LIST TABLES [container.]owner.table [AUTO_CAPTURE]
```

table
The name of a table or a group of tables specified with a wildcard (*).

[*container.*]*owner.table*
The command accepts a two-part name in non-CDB mode and a three-part name in CDB mode. Supported wildcards are ? and * for single and zero or more character matching respectively. In Admin Client, both the schema/owner and container name cannot contain a

wildcard character. Table name can have wildcards. When the command is successful, the output shows the list of tables enabled for auto capture matching the input criteria.

Example
The following shows a `LIST TABLES` command and sample output.

```
LIST TABLES empl*
```

Example
The following example shows a `LIST TABLES` command listing tables enabled for auto capture:

```
LIST TABLES hr.emp*  AUTO_CAPTURE
```

The output is similar to the following:

```
2022-09-06T06:31:09Z  INFO    OGG-15189  Default catalog name DBEAST will be
used for table specification hr.emp*.
"DBEAST"."HR"."EMPLOYEES"
"DBEAST"."HR"."EMP_DETAILS_VIEW"

Found 2 tables matching list criteria.
```

MININGDBLOGIN

Use `MININGDBLOGIN` to establish a connection to a downstream Oracle AI Database logmining server in preparation to issue other Oracle GoldenGate commands that affect this database, such as `REGISTER EXTRACT`.

To log into a source Oracle AI Database that serves as the database logmining server, use the `DBLOGIN` command. `MININGDBLOGIN` is reserved for login to a downstream mining database.

The user who issues `MININGDBLOGIN` must:

- have privileges granted through the Oracle `dbms_goldengate_auth.grant_admin_privilege` procedure.
- be the user that is specified with the `TRANLOGOPTIONS MININGUSER` parameter for the Extract group that is associated with this `MININGDBLOGIN`.

Syntax

```
MININGDBLOGIN USERIDALIAS alias [DOMAIN domain]
```

`USERIDALIAS alias [DOMAIN domain]`
Supplies the alias of a database login credential. Can be used instead of the `USERID` option if there is a local Oracle GoldenGate credential store that contains a credential with the required privileges for this `MININGDBLOGIN` command.
To log into a pluggable database in an Oracle multitenant container database, the user must be stored as a connect string, such as `OGGUSER@FINANCE`. To log into the root container, the user must be stored as a common user, including the c## prefix, such as `c##ggadmin@ggnorth`.

alias
Specifies the alias of a database user credential that is stored in the Oracle GoldenGate credential store. The user that is specified with USERIDALIAS must be the common database user.

DOMAIN *domain*
Specifies the credential store domain for the specified alias. A valid domain entry must exist in the credential store for the specified alias.

SYSDBA
(Oracle) Specifies that the user logs in as sysdba. This option can be used for USERID and USERIDALIAS.

Examples

```
MININGDBLOGIN USERIDALIAS ggeast
```

```
MININGDBLOGIN USERID ggadmin@pdbeast.example.com, PASSWORD
AACAAAAAAAAAAAJAUEUGODSCVGJEEIUGKJDJTFNDKEJFFFTC AES128, ENCRYPTKEY securekey1
```

NOALLOWNESTED

Use the NOALLOWNESTED command to disable the use of nested OBEY files. A nested OBEY file is one that references another OBEY file. This is the default setting for OBEY files.

Syntax:

```
NOALLOWNESTED
```

When you exit your Admin Client session, the next Admin Client session reverts to NOALLOWNESTED. This is the default. An attempt to run a nested OBEY file in the default mode of NOALLOWNESTED causes an error that is similar to the following:

```
ERROR: Nested OBEY scripts not allowed. Use ALLOWNESTED to allow nested
scripts.
```

OBEY

Use OBEY to process a file that contains a list of Oracle GoldenGate commands. OBEY is useful for executing commands that are frequently used in sequence.

You can call one OBEY file from another one. This is called a nested OBEY file. You can nest up to 16 OBEY files. To use nested OBEY files, you must enable the functionality by first issuing the ALLOWNESTED command, see ALLOWNESTED.

Syntax

```
OBEY file_name
```

file_name
The relative or fully qualified path name of the file that contains the list of commands.

Examples

```
OBEY ./mycommands.txt
```

```
ADD EXTRACT exte, TRANLOG, BEGIN NOW
add exttrail east/ea, EXTRACT exte
ADD EXTRACT extw, TRANLOG, BEGIN NOW
ADD EXTRACT west/ew, EXTRACT extw
ADD REPLICAT repe, EXTTRAIL east/ea, BEGIN NOW
ADD REPLICAT repw, EXTTRAIL west/ew, BEGIN NOW
```

The preceding command executes the `mycommands.txt` file and dispalys its content with the `ADD` commands.
The following example displays the content of the `startcmds.txt` file.

```
OBEY ./startcmds.txt
```

```
START EXTRACT *
INFO EXTRACT *, DETAIL
START REPLICAT *
INFO REPLICAT *, DETAIL
```

PURGE EXTTRAIL

Use `PURGE EXTTRAIL` to remove files related to a local trail from the file system. Partial files are not deletes.

Syntax

```
PURGE EXTTRAIL trail-name | !
```

trail-name
The relative or fully qualified path name of the trail.

!
(Exclamation point) Use to purge a trail or distribution path that is in use by an Extract.

Example:

For example, if the trails files are:

```
a2000000001
a2000000002
a2000000003
a2000000004
```

And Extract is using the a2000000004 trail, none of the trail files are not purged. You can only purge these files after the Extract is deleted and no longer using any of the files.

PURGE WALLET

Use the PURGE WALLET command to permanently remove master key versions from the master-key wallet. Only the versions that are marked for deletion by the DELETE MASTERKEY command are removed. The purge is not reversible.

> **Note**
>
> For Oracle GoldenGate deployments using a shared wallet, the older versions of the master key should be retained after the master key is renewed until all processes are using the newest version. The time to wait depends on the topology, latency, and data load of the deployment. A minimum wait of 24 hours is a conservative estimate, but you may need to perform testing to determine how long it takes for all processes to start using a new key. To determine whether all of the processes are using the newest version, view the report file of each Extract immediately after renewing the master key to confirm the last SCN that was mined with the old key. Then, monitor the Replicat report files to verify that this SCN was applied by all Replicat groups. At this point, you can delete the older versions of the master key.

The OPEN WALLET command must be used before using this command or any of the commands that add, renew, or delete the master keys in the wallet.

After purging a wallet that is not maintained centrally on shared storage, the updated wallet can be copied to all of the other systems in the Oracle GoldenGate configuration that use this wallet, so that no purged keys remain in the configuration. Before doing so, Extract must be stopped and then all of the downstream Oracle GoldenGate processes must be allowed to finish processing their trails and then be stopped. After the wallet is copied into place, the processes can be started again.

Syntax

```
PURGE WALLET
```

REGISTER EXTRACT

This command applies to Oracle AI Database and PostgreSQL.

Oracle AI Database

This command registers a primary Extract group to enable integrated capture mode and specify options for integrated Extract from a multitenant container database

To unregister an Extract group from the database, use the UNREGISTER EXTRACT command.

Syntax:

```
REGISTER EXTRACT group-name
          ( , DATABASE
          ( [ CONTAINER container-list |
              ADD   CONTAINER container-list |
```

```
    DROP  CONTAINER container-list ]
  [ SCN   scn   ]
  [ SHARE ( AUTOMATIC | group-name | NONE ) ]
  [ [NO]OPTIMIZED ]
)
```

Container-list is a comma separated list of PDB names, for example `(pdbeast, pdbwest)`; or wildcarded PDB names, for example `(pdb* or pdb?)`; or both, for example `(cdbnorth, pdb*)`. Supported wildcards are `?` and `*`.

The `OPTIMIZED` option improves Extract fast startup. The default value is `NOOPTIMIZED`. The `OPTIMIZED` option only impacts an upstream non multitenant configuration.

DATABASE [
CONTAINER (*container*[, ...]) |
ADD CONTAINER (*container*[, ...]) |
DROP CONTAINER (*container*[, ...])
]
Valid for Oracle.
Without options, `DATABASE` enables integrated capture from a non-CDB database for the Extract group. In this mode, Extract integrates with the database logmining server to receive change data in the form of logical change records (LCR). Extract does not read the redo logs. Extract performs capture processing, transformation, and other requirements. The DML filtering is performed by the logmining server.
Before using `REGISTER EXTRACT` with `DATABASE`, use the DBLOGIN USERIDALIAS command for all Extracts with the privileges granted using the `dbms_goldengate_auth.grant_admin_privilege` procedure. If you have a downstream configuration, then you must also issue the MININGDBLOGIN command. If the source database you are registering is a CDB database and Extract will fetch data, then `grant_admin_privilege` must be called with the `CONTAINER=>'ALL'` parameter.
After using `REGISTER EXTRACT`, use `ADD EXTRACT` with the `INTEGRATED TRANLOG` option to create an Extract group of the same name.

CONTAINER (*container*[, ...])
Applies the registration to a list of one or more pluggable databases (containers) of a multitenant container database (CDB). Specify one or more pluggable databases as a comma-delimited list within parentheses, for example: `CONTAINER (pdbeast, pdbwest)`. If you list the pluggable databases, they must exist in the database. You can also specify the pluggable databases using the wildcards `*` and `?`. For example, `CONTAINER (pdb*)`.

ADD CONTAINER (*container*[, ...])
Adds the specified pluggable database to an existing Extract capture configuration. Specify one or more pluggable databases as a comma-delimited list within parentheses or using the wildcards `*` and `?`. For example: `ADD CONTAINER (pdbeast, pdbwest)`. Before issuing `REGISTER EXTRACT` with this option, stop the Extract group.
For Oracle, adding containers at particular SCN on an existing Extract is not supported.

DROP CONTAINER (*container*[, ...])
Drops the specified pluggable database from an existing Extract capture configuration. Specify one or more pluggable databases as a comma-delimited list within parentheses or using the wildcards `*` and `?`. For example,`DROP CONTAINER (pdbeast, pdbwest)`.
Registering the Extract after running the `drop container` option, does not fully happen until the Extract has been started and it reads a committed transaction from a dropped pluggable database, which is greater than the Extract checkpoint SCN. Extract then fully drops the containers and shuts down with a message.

Before running `REGISTER EXTRACT` with this option, stop the Extract group.

`SCN system_change_number`
Registers Extract to begin capture at a specific system change number (SCN) in the past. Without this option, capture begins from the time that `REGISTER EXTRACT` is issued. The specified SCN must correspond to the begin SCN of a dictionary build operation in a log file. You can issue the following query to find all valid SCN values from dictionary dumps at CDB level:

```
SELECT first_change#
   FROM v$archived_log
   WHERE dictionary_begin = 'YES' AND
      standby_dest = 'NO' AND
      name IS NOT NULL AND
      status = 'A';
```

However, this query does not work when registering a per-PDB Extract. To identify start SCN of available dictionary dumps for a specific PDB, we can follow these steps:

1. Connect to the PDB.
2. Run the `show con_id` command to get the PDB identifier.
3. Run the following query to list all available dictionary dumps in descending order:

   ```
   SELECT DATE_OF_BUILD, START_SCN FROM dba_logmnr_dictionary_buildlog WHERE
   CONTAINER_ID = PDB identifier ORDER BY DATE_OF_BUILD DESC;
   ```

When used alone, the SCN value is the beginning SCN of the dictionary build operation in a log file.
When used in conjunction with `SHARE AUTOMATIC` or `SHARE` *`extract_name`*, then the specified SCN is the `start_scn` for the capture session and has the following restrictions:

- Should be lesser than or equal to the current SCN.
- Should be greater than the minimum (first SCN) of the existing captures.

`{SHARE [`
`AUTOMATIC |`
***`extract`*`|`**
`NONE]}`
Valid for Oracle.
Registers the Extract to return to an existing LogMiner data dictionary build with a specified SCN creating a clone. This allows for faster creation of Extracts by leveraging existing dictionary builds.
`SHARE` cannot be used on a CDB.
The following commands are supported:

```
REGISTER EXTRACT extract database SCN #### SHARE AUTOMATIC
REGISTER EXTRACT extract database SCN #### SHARE extract
REGISTER EXTRACT extract database SHARE NONE
REGISTER EXTRACT extract database SCN #### SHARE NONE
```

Or

```
REGISTER EXTRACT extract DATABASE SHARE NONE
REGISTER EXTRACT extract DATABASE SCN #### SHARE NONE
```

In contrast, the following commands are *not* supported in a downstream configuration:

```
REGISTER EXTRACT extract DATABASE SHARE AUTOMATIC
REGISTER EXTRACT extract DATABASE SHARE extract
```

AUTOMATIC
Clone from the existing closest capture. If no suitable clone candidate is found, then a new build is created.

extract
Clone from the capture session associated for the specified Extract. If this is not possible, then an error occurs the register does not complete.

NONE
Does not clone or create a new build; this is the default.

In a downstream configuration, the `SHARE` clause *must* be used in conjunction with the `SCN` clause when registering for Extract.

PostgreSQL

This command creates a replication slot in the connected source database and ensures that the PostgreSQL database does not purge the transaction log until the replication slot is moved or removed. The `REGISTER EXTRACT` command must be run before running the `ADD EXTRACT` command.

You must connect to the PostgreSQL database instance using `DBLOGIN USERIDALIAS` before registering the Extract.

You can also register an Extract from the Oracle GoldenGate Administration Service web interface. For details, see Register Extract for PostgreSQL in the *Oracle GoldenGate Microservices Documentation*.

You can also set up streaming options for PostgreSQL, using the `TRANLOGOPTIONS STREAMINGOPTIONS` parameter, to supply parameters during replication startup. For details, see `TRANLOGOPTIONS`,

> **Note**
>
> Starting with Oracle GoldenGate 21c (21.3) release onward, it's not mandatory to enter the *`database_name`*.

Syntax:

```
REGISTER EXTRACT group_name
```

or

```
REGISTER EXTRACT group_name PGPLUGINTYPE pgoutput;
```

> **Note**
>
> To explicitly register with the `pgoutput` plugin, you need to specify the `pgoutput` plugin type in the command. If you do not select `pgoutput` plugin, then the default plugin type, `test_decoding`, is set.

or

```
REGISTER EXTRACT group_name MIGRATE replication_slot_name
```

or

```
REGISTER EXTRACT group_name with DATABASE database_name
```

or

group_name
The name of the Extract group that is to be registered. Do not use a wildcard.

`MIGRATE` *`replication_slot_name`*
Valid for PostgreSQL.
In case of Oracle GoldenGate upgrade, where the installation directory is different for the Extract processes, `REGISTER EXTRACT` with the `MIGRATE` option is required.
The *`replication_slot_name`* is the name of any existing replication slot present in the database. This command creates a new replication slot by copying the LSN information present in the original slot *`replication_slot_name`*. The copied replication slot starts from the same LSN as the original one.

> **Note**
>
> The original replication slot must be present in the same database from where we want to create a new slot.

`PGPLUGINTYPE test_decoding | pgoutput`
Valid for PostgreSQL.
This option has been added to allow selecting the logical decoding plugin for PostgreSQL. You can select `test_decoding` or `pgoutput`. By default, the CDC Extract registers using the `test_decoding` plugin type. So, if you don't specify the `pgplugintype` as `pgoutput`, then `test_decoding` will be used automatically.

Examples

```
REGISTER EXTRACT exte LOGRETENTION
```

```
REGISTER EXTRACT exte DATABASE
```

```
REGISTER EXTRACT exte DATABASE CONTAINER (pdbeast, pdbwest, pdbsouth)
```

```
REGISTER EXTRACT exte DATABASE ADD CONTAINER (pdbnorth)
```

```
REGISTER EXTRACT exte DATABASE DROP CONTAINER (pdbnorth)
```

```
REGISTER EXTRACT exte DATABASE SCN 136589
```

The beginning SCN of the dictionary build is 136589.

```
REGISTER EXTRACT exte DATABASE SCN 67000 SHARE extw
```

The valid start SCN, 67000 in this case; it is not necessarily the current SCN.

```
REGISTER EXTRACT exte DATABASE CONTAINER (pdbeast, pdbeast, pdbsouth) SCN
136589
```

For PostgreSQL with the plugin type specified as `PGOUTOPUT`:

```
REGISTER EXTRACT exte PGPLUGINTYPE pgoutput WITH DATABASE hr
```

RENEW MASTERKEY

Use the `RENEW MASTERKEY` command to create a new version of the master encryption key in the master-key wallet. The key name remains the same, but the bit ordering is different. All versions of a master key remain in the wallet until they are marked for deletion with the `DELETE MASTERKEY` command and then the wallet is purged with the `PURGE WALLET` command.

The `OPEN WALLET` command must be used before using this command or any of the commands that add or delete the master keys or purge the wallet.

After renewing a master key in a wallet that is not maintained centrally on shared storage, the updated wallet must be copied to all of the other systems in the Oracle GoldenGate configuration that use this wallet. Before doing so, Extract must be stopped and then all of the downstream Oracle GoldenGate processes must be allowed to finish processing their trails and then be stopped. After the wallet is copied into place, the processes can be started again..

Syntax

```
RENEW MASTERKEY
```

Example

```
RENEW MASTERKEY
```

RESTART DEPLOYMENT

Use `RESTART DEPLOYMENT` to restart the specified deployment.

Syntax

```
RESTART DEPLOYMENT deployment-name-wildcard
```

`deployment-name-wildcard`
The name of the deployment or a wildcard (`*`) to specify multiple deployments. For example, `P*` restarts all deployments whose names begin with P.

Example

```
RESTART DEPLOYMENT NORTH
```

RESTART ER

Use `RESTART ER` to stop then start the specified wildcarded groups. ER processes that are already stopped are started.

Syntax

```
RESTART ER group-name-wildcard
```

The name of the group or a wildcard (`*`) to specify multiple groups. For example, `T*` restarts all groups whose names start with T.

RESTART EXTRACT

Use `RESTART EXTRACT` to stop then start an Extract group.

Syntax

```
RESTART EXTRACT group_name [ATCSN csn | AFTERCSN csn]
```

`group_name`
The name of an Extract group or a wildcard (`*`) to specify multiple groups. For example, `T*` starts all Extract groups whose names begin with T.

`ATCSN` *`csn`* `|` `AFTERCSN` *`csn`*
Specifies an alternate start point.

ATCSN
Directs Extract to position its start point at the first transaction that has the specified CSN. Any transactions in the data source that have CSN values less than the specified one are skipped.

AFTERCSN
Directs Extract to position its start point at the beginning of the first transaction after the one that has the specified CSN. Any transactions in the data source that have CSN values that are less than, or equal to, the specified one are skipped.

csn
Specifies a CSN value. Enter the CSN value in the format that is valid for the database. Extract abends if the format is invalid and writes a message to the report file. To determine the CSN to supply after an initial load is complete, use the serial identifier at which the load utility completed. Otherwise, follow the instructions in the initial load procedure for determining when to start Extract.

The following are additional guidelines to observe when using `ATCSN` and `AFTERCSN`:

- The CSN is stored in the file header so that it is available to downstream processes.
- When a record that is specified with a CSN is found, Extract issues a checkpoint. The checkpoint ensures that subsequent Extract start ups begin from the requested location, and not from a point prior to the requested CSN.
- You must establish a physical start point in the transaction log or trail for Extract with `ADD EXTRACT` or `ALTER EXTRACT` before using `ATCSN` or `AFTERCSN`. These options are intended to be an additional filter after Extract is positioned to a physical location in the data source.

Examples

```
RESTART EXTRACT finance
```

```
RESTART EXTRACT finance ATCSN 684993
```

```
RESTART EXTRACT finance AFTERCSN 684993
```

RESTART REPLICAT

Use `RESTART REPLICAT` to stop then start a Replicat group. To confirm that Replicat has started, use the `INFO REPLICAT` or `STATUS REPLICAT` command.

Normal Start Point

Replicat can be started at its normal start point (from initial or current checkpoints) or from an alternate, user-specified position in the trail.

`RESTART REPLICAT`, without any options, causes Replicat to start processing at one of the following points to maintain data integrity:

- After graceful or abnormal termination: At the first unprocessed transaction in the trail from the previous run, as represented by the current read checkpoint.

- First-time startup after the group was created: From the beginning of the active trail file (`seqno 0`, `rba 0`).

Alternate Start Point

The `SKIPTRANSACTION`, `ATCSN`, and `AFTERCSN` options of `START REPLICAT` cause Replicat as a whole, or specific threads of a coordinated Replicat, to begin processing at a transaction in the trail other than the normal start point. Use these options to:

- Specify a logical recovery position when an error prevents Replicat from moving forward in the trail. Replicat can be positioned to skip the offending transaction or transactions, with the understanding that the data will not be applied to the target.
- Skip replicated transactions that will cause duplicate-record and missing-record errors after a backup is applied to the target during an initial load. These options cause Replicat to discard transactions that occurred earlier than the most recent set of changes that were captured in the backup. You can map the value of the serial identifier that corresponds to the completion of the backup to a CSN value, and then start Replicat to begin applying transactions from the specified CSN onward.

Syntax

```
RESTART REPLICAT group_name_wildcard
[SKIPTRANSACTION | {ATCSN csn | AFTERCSN csn}]
[FILTERDUPTRANSACTIONS | NOFILTERDUPTRANSACTIONS]
[THREADS (threadID[, threadID][, ...][, thread_range[, thread_range][, ...])
```

group_name_wildcard
The name of a Replicat group or a wildcard (*) to specify multiple groups. For example, `T*` starts all Replicat groups whose names begin with T.

`SKIPTRANSACTION`
Causes Replicat to skip the first transaction after its expected startup position in the trail. All operations from that first transaction are excluded.
If the `MAXTRANSOPS` parameter is also being used for this Replicat, it is possible that the process will start to read the trail file from somewhere in the middle of a transaction. In that case, the remainder of the partial transaction is skipped, and Replicat resumes normal processing from the next begin-transaction record in the file. The skipped records are written to the discard file if the `DISCARDFILE` parameter is being used; otherwise, a message is written to the report file that is similar to:

```
User requested START SKIPTRANSACTION. The current transaction will be
skipped. Transaction ID txid, position Seqno seqno, RBA rba
```

`SKIPTRANSACTION` is valid only when the trail that Replicat is reading is part of an online change synchronization configuration (with checkpoints). Not valid for task-type initial loads (where `SPECIALRUN` is used with `ADD REPLICAT`).

ATCSN *csn* | AFTERCSN *csn*
Sets a user-defined start point at a specific CSN. When ATCSN or AFTERCSN is used, a message similar to one of the following is written to the report file:

```
User requested start at commit sequence number (CSN) csn-string
```

```
User requested start after commit sequence number (CSN) csn-string
```

General information about these options:

- Valid only when the trail that Replicat is reading is part of an online change synchronization configuration (with checkpoints). Not valid for task-type initial loads (where SPECIALRUN is used with ADD REPLICAT).
- To support starting at, or after, a CSN, the trail must be of Oracle GoldenGate version 10.0.0 or later, because the CSN is stored in the first trail record of each transaction. If Replicat is started with AFTERCSN against an earlier trail version, Replicat will abend and write an error to the report stating that the trail format is not supported.

ATCSN
Causes Replicat to start processing at the transaction that has the specified CSN. Any transactions in the trail that have CSN values that are less than the specified one are skipped.

AFTERCSN
Causes Replicat to start processing at the transaction that occurred after the one with the specified CSN. Any transactions in the trail that have CSN values that are less than, or equal to, the specified one are skipped.

csn
Specifies a CSN value. Enter the CSN value in the format that is valid for the database. See Commit Sequence Number (CSN) for CSN formats and descriptions. Replicat abends if the format is invalid and writes a message to the report file.

To determine the CSN to supply after an initial load is complete, use the commit identifier at which the load utility completed the load. Otherwise, follow the instructions in the initial load procedure for determining when to start Replicat.

FILTERDUPTRANSACTIONS | NOFILTERDUPTRANSACTIONS
Causes Replicat to ignore transactions that it has already processed. Use when Extract was repositioned to a new start point (see the ATCSN or AFTERCSN option of "START EXTRACT") and you are confident that there are duplicate transactions in the trail that could cause Replicat to abend. This option requires the use of a checkpoint table. If the database is Oracle, this option is valid only for Replicat in non-integrated mode. In case of Integrated mode and automatic target trail file regeneration, the Integrated mode handles the duplicate transactions transparently. The default is FILTERDUPTRANSACTIONS.

THREADS *thread_list*
Valid for SKIPTRANSACTION, ATCSN, and AFTERCSN when Replicat is in coordinated mode. Not valid for RESTART REPLICAT without those options. Starts the specified Replicat thread or threads at the specified location.

thread_list
A comma-delimited list of ranges in the format of threadIDlow-threadIDhigh, threadIDlow-threadIDhigh.

!
(Exclamation point) Restarts Replicat immediately. The transaction is stopped.

Examples

```
RESTART REPLICAT finance
```

The following starts Replicat at a-specific CSN.

```
RESTART REPLICAT finance, ATCSN 6488359
```

The following causes threads 4 and 5 of a coordinated Replicat to skip the first transaction after their last checkpoint when Replicat is started. If this were a 10-thread coordinated Replicat, threads 0-3 and 6-10 would all start at the normal start point, that of their last checkpoint.

```
RESTART REPLICAT fin SKIPTRANSACTION THREADS (4-5)
```

The following example causes threads 1-3 of a coordinated Replicat to start at CSN 6488359, threads 9-10 to start after CSN 6488360, and threads 7 and 8 to skip the first transaction after its last checkpoint.

```
RESTART REPLICAT fin ATCSN 6488359 THREADS (1-3), AFTERCSN 6488360 THREADS
(9-10), SKIPTRANSACTION THREADS (7,8)
```

RESTART SERVICE

Use `RESTART SERVICE` to restart the specified Oracle GoldenGate services.

Syntax

```
RESTART SERVICE service-name-wildcard
```

service-name-wildcard
The name of an service or a wildcard (*) to specify multiple services. Valid services are `ADMINSRVR`, `DISTSRVR`, `RECVSRVR`, and `PMSRVR`.

Example

```
RESTART SERVICE ADMIN*
```

SEND ER

Use the `SEND ER` to get send information on multiple Extract and Replicat groups as a unit. Use it with wildcards to affect every Extract and Replicat group that satisfies the wildcard.

Syntax

```
SEND ER group_name
```

group_name
The wildcard specification for the groups that you want to affect with the command.
The following example sends a status request for the Replicat `repe` and receives the response that it's not currently running.

Example

```
SEND ER rep* STATUS
Sending STATUS request to REPLICAT repe ...
2020-01-29 14:30:11
ERROR   OGG-15148  REPLICAT repe not currently running.
```

SEND EXTRACT

Use `SEND EXTRACT` to communicate with a running Extract process. The request is processed as soon as Extract is ready to accept commands from users.

Syntax

```
SEND EXTRACT group_name, {
BR {BRINTERVAL interval |
   BRSTART |
   BRSTOP |
   BRCHECKPOINT {IMMEDIATE | IN n{M|H} | AT yyyy-mm-dd hh:mm[:ss]]}} |
BR BRFSOPTION { MS_SYNC | MS_ASYNC }
BR BRSTATS
BR BRSTATUS
CACHEMGR {CACHESTATS {ALL | MINIMAL | SUPERPOOL | CURRENTONLY | POOLS | POOL
n | QUEUES} | CACHEQUEUES | CACHEVMUSAGE} |
FORCESTOP |
FORCETRANS transaction_ID [FORCE] |
GETLAG |
GETPARAMINFO [parameter_name] [FILE output_file] |
GETTCPSTATS |
LOGEND |
LOGSTATS |
REPORT |
RESUME |
ROLLOVER |
SHOWTRANS [transaction_ID] [COUNT n]
    [DURATION duration unit] [TABULAR]
    [FILE file_name [DETAIL]] [ALL]|
SKIPTRANS transaction_ID [FORCE] |
STATUS |
STOP |
TRACE[2] file_name |
TRACE[2] OFF |
TRACE OFF file_name |
TRACEINIT |
TRANLOGOPTIONS INTEGRATEDPARAMS(parameter_specification) |
TRANLOGOPTIONS {TRANLOGOPTIONS {PURGEORPHANEDTRANSACTIONS |
NOPURGEORPHANEDTRANSACTIONS} |
TRANLOGOPTIONS TRANSCLEANUPFREQUENCY minutes |
VAMMESSAGE 'Teradata_command' |
```

```
VAMMESSAGE {'ARSTATS' | 'INCLUDELIST [filter]' | 'FILELIST [filter]'|
'EXCLUDELIST [filter]'} |
VAMMESSAGE 'OPENTRANS'
}
```

group_name
The name of the Extract group or a wildcard (*) to specify multiple groups. For example, T* sends the command to all Extract processes whose group names start with T. If an Extract is not running, an error is returned.

BR {BRINTERVAL *interval* | BRSTART | BRSTOP |
BRCHECKPOINT {IMMEDIATE | IN *n* {H|M} | AT yyyy-mm-dd[hh:mm[:ss]]}}
Sends commands that affect the Bounded Recovery mode of Extract.

BRINTERVAL *interval*
Sets the time between Bounded Recovery checkpoints. Valid values are from 20 minutes to 96 hours specified as `M` for minutes or `H` for hours, for example `20M` or `2H`. The default interval is 4 hours.

BRSTART
Starts Bounded Recovery. This command should only be used under direction of Oracle Support.

BRSTOP
Stops Bounded Recovery for the run and for recovery. Consult Oracle Support before using this option. In most circumstances, when there is a problem with Bounded Recovery, it turns itself off.

BRCHECKPOINT {IMMEDIATE | IN *n*{H|M} | AT *yyyy-mm-dd[hh:mm*[*:ss*]]}}
Sets the point at which a bounded recovery checkpoint is made. `IMMEDIATE` issues the checkpoint immediately when `SEND EXTRACT` is issued. `IN` issues the checkpoint in the specified number of hours or minutes from when `SEND EXTRACT` is issued. `AT` issues the checkpoint at exactly the specified time.

BR BRFSOPTION {MS_SYNC | MS_ASYNC}
Performs synchronous/asynchronous writes of the mapped data in Bounded Recovery.

MS_SYNC
Bounded Recovery writes of mapped data are synchronized for I/O data integrity completion.

MS_ASYNC
Bounded Recovery writes of mapped data are initiated or queued for servicing.

BR BRSTATS
Provides details on each of the object pools, which were persisted by BR. This includes the following details:

- Object sizes through life of the Extract group
- Object ages through the life of Extract group
- Extant object sizes as of the most recent BCP
- Extant object ages as of the most recent BCP

BR BRSTATUS
Returns status for the Bounded Recovery mode of Extract.

It shows the following:

- Current settings of the BR parameter.
- Current status of the Bounded Recovery, if one was performed, with current checkpoint interval, timestamps for the next and last checkpoints, and the total and outstanding number of objects and sizes when BR is in progress.
- Start and end sequence, RBA, SCN, and timestamp for the Bounded Recovery checkpoint positions (per redo thread).

CACHESTATS {ALL | MINIMAL | SUPERPOOL | CURRENTONLY | POOLS | POOL n | QUEUES}
CACHESTATS returns statistics for virtual memory usage and file caching.
ALL
Returns all COM statistics and is the default option.
MINIMAL
Returns condensed (minimal) version of COM stats whereas CACHEALL returns all statistics.
SUPERPOOL
Super pool statistics. Returns statistics about all object pools.
CURRENTONLY
Returns statistics only for the run time.
POOLS
Returns statistics for all object pools.
POOL *n*
Returns statistics for the specified object pool only.
QUEUES
Returns statistics for queues only.
CACHEVMUSAGE
Returns statistics for the virtual memory use.

FORCESTOP
Forces Extract to stop, bypassing any notifications. This command will stop the process immediately.

FORCETRANS *transaction_ID*] [FORCE]
Valid for Db2 LUW, Db2 IBM, MySQL, Oracle, and SQL Server.
Forces Extract to write a transaction specified by its transaction ID number to the trail as a committed transaction. FORCETRANS does not commit the transaction to the source database. It only forces the existing data to the trail so that it is processed (with an implicit commit) by Replicat. You can repeat FORCETRANS for other transactions in order of their age. Note that forcing a transaction to commit to the trail (and therefore the target database) may cause data discrepancies if the transaction is rolled back by the source user applications.
After using FORCETRANS, wait at least five minutes if you intend to issue SEND EXTRACT with FORCESTOP. Otherwise, the transaction will still be present.
If FORCETRANS is used immediately after Extract starts, you might receive an error message that asks you to wait and then try the command again. This means that no other transactions have been processed yet by Extract. Once another transaction is processed, you will be able to force the transaction to trail.

transaction_ID
The ID of the transaction. Get the transaction ID number with SHOWTRANS or from an Extract runtime message. Extract ignores any data added to the transaction after this command is issued. A confirmation prompt must be answered unless FORCE is used. To use FORCETRANS, the specified transaction must be the oldest one in the list of transactions shown with SHOWTRANS.

FORCE
Valid for Oracle and SQL Server. Not valid for MySQL.
Use `FORCE` to bypass the confirmation prompt.

GETLAG
Determines a true lag time between Extract and the data source. Returns the same results as LAG EXTRACT.

GETPARAMINFO [*parameter_name*] [FILE *output_file*]
Use `GETPARAMINFO` to query runtime parameter values of a running instance, including Extract, Replicat, and Manager. You can query for a single parameter or all parameters and send the output to the console or a text file

parameter_name
The default behavior is to display all parameters in use, meaning those parameters that have ever been queried by the application, parameters, and their current values. If you specify a particular parameter, then the output is filtered by that name.

FILE *output_file*
The name of the text file that your output is redirected to.

GETTCPSTATS
Displays statistics about network activity between Extract and the target system. The statistics include:

- Local and remote IP addresses.
- Inbound and outbound messages, in bytes and bytes per second.
- Number of receives (inbound) and sends (outbound). There will be at least two receives per inbound message: one for the length and one or more for the data.
- Average bytes per send and receive.
- Send and receive wait time: Send wait time is how long it takes for the write to TCP to complete. The lower the send wait time, the better the performance over the network. Receive wait time is how long it takes for a read to complete. Together, the send and receive wait times provide a rough estimate of network round trip time. These are expressed in microseconds.
- Status of data compression (enabled or not).
- Uncompressed bytes and compressed bytes: When compared (uncompressed to compressed), these comprise the compression ratio, meaning how many bytes there were before and after compression. You can compare the compression ratio with the bytes that are being compressed per second to determine if the compression rate is worth the cost in terms of resource and network consumption.

The `TCPBUFSIZE` option of `RMTHOST` and `RMTHOSTOPTIONS` controls the size of the TCP buffer for uncompressed data. What actually enters the network will be less than this size if compression is enabled. `GETTCPSTATS` shows post-compression throughput.

LOGEND
Confirms whether or not Extract has processed all of the records in the data source.

LOGSTATS
Valid only for Oracle.
Instructs Extract to issue a report about the statistics that are related to the processing of data from the Oracle redo log files. Extract uses an asynchronous log reader that reads ahead of the current record that Extract is processing, so that the data is available without additional I/O

on the log files. The processing is done through a series of read/write queues. Data is parsed by a producer thread at the same time that additional data is being read from the log file by a reader thread. Thus, the reason for the term "read-ahead" in the statistics.
The statistics are:

- `AsyncReader.Buffers`*n*: There is a field like this for each buffer queue that contains captured redo data. It shows the size, the number of records in it, and how long the wait time is before the data is processed. These statistics are given for write operations and read operations on the queue.
- `REDO` read ahead buffers: The number of buffers that are being used to read ahead asynchronously.
- `REDO` read ahead buffer size: The size of each buffer.
- `REDO` bytes read ahead for current redo: Whether read-ahead mode is on or off for the current redo log file (value of `ON` or `OFF`).
- `REDO` bytes read: The number of bytes read from all redo log files that are associated with this instance of Extract.
- `REDO` bytes read ahead: The number of bytes that were processed by the read-ahead mechanism.
- REDO bytes unused: The number of read-ahead bytes that were subsequently dropped as the result of Extract position changes or stale reads.
- `REDO` bytes parsed: The number of bytes that were processed as valid log data.
- `REDO` bytes output: The number of bytes that were written to the trail file (not including internal Oracle GoldenGate overhead).

`REPORT`
Generates an interim statistical report to the Extract report file. The statistics that are displayed depend upon the configuration of the `STATOPTIONS` parameter when used with the `RESETREPORTSTATS` | `NORESETREPORTSTATS` option.

`RESUME`
Resumes (makes active) a process that was suspended by an `EVENTACTIONS SUSPEND` event. The process resumes normal processing from the point at which it was suspended.

`ROLLOVER`
Causes Extract to increment to the next file in the trail when restarting. For example, if the current file is `ET000002`, the current file will be `ET000003` after the command executes. A trail can be incremented from `000001` through `999999`, and then the sequence numbering starts over at `000000`.

`SHOWTRANS [`*`transaction_ID`*`] [COUNT `*`n`*`]`
`[DURATION `*`duration unit`*`] [TABULAR] | [FILE `*`file_name`*` [DETAIL]] [ALL]`
Valid for Db2 IBM, Db2 LUW, Db2 z/OS, MySQL, Oracle, and SQL Server.
Displays information about open transactions. `SHOWTRANS` shows any of the following, depending on the database type:

- Process checkpoint (indicating the oldest log needed to continue processing the transaction in case of an Extract restart).
- Transaction ID
- Extract group name
- Redo thread number

- Timestamp of the first operation that Oracle GoldenGate extracts from a transaction (not the actual start time of the transaction)
- System change number (SCN)
- Redo log number and RBA
- Status (`Pending COMMIT` or `Running`). `Pending COMMIT` is displayed while a transaction is being written after a `FORCETRANS` was issued.

Without options, `SHOWTRANS` displays all open transactions that will fit into the available buffer. However, it doesn't display the output user name sometimes for an open active transaction because the user name is not provided in the begin record from transaction log.
See the examples for sample output of `SHOWTRANS`. To further control output, see the following options.

transaction_ID
Limits the command output to a specific transaction.

COUNT *n*
Constrains the output to the specified number of open transactions, starting with the oldest one. Valid values are `1` to `1000`.

DURATION *duration unit*
Restricts the output to transactions that have been open longer than the specified time, where:
duration is the length of time expressed as a whole number.
unit is one of the following to express `seconds`, `minutes`, `hours`, or `days`:

```
S|SEC|SECS|SECOND|SECONDS
M|MIN|MINS|MINUTE|MINUTES
H|HOUR|HOURS
D|DAY|DAYS
```

TABULAR
Valid only for Oracle.
Generates output in tabular format similar to the default table printout from SQL*Plus. The default is field-per-row.

FILE *file_name* [DETAIL]
Valid only for Oracle and SQL Server. Not valid for MySQL.
Forces Extract to write the transaction information to the specified file. There is no output to the console.
For Oracle, you can write a hex and plain-character dump of the data by using `FILE` with `DETAIL`. This dumps the entire transaction from memory to the file. Viewing the data may help you decide whether to skip the transaction or force it to the trail.

> **Note**
>
> Basic detail information is automatically written to the report file at intervals specified by the `WARNLONGTRANS CHECKINTERVAL` parameter.

[ALL]
This option allows showing all the transaction in COM. It is useful when detecting potential issues with committed transactions that are remaining in COM.

SKIPTRANS *transaction_ID* [FORCE]
Valid for Db2 LUW, Db2 IBM, MySQL, Oracle, and SQL Server.
Forces Extract to skip the specified transaction, thereby removing any current data from memory and ignoring any subsequent data. A confirmation prompt must be answered unless `FORCE` is used. After using `SKIPTRANS`, wait at least five minutes if you intend to issue `SEND EXTRACT` with `FORCESTOP`. Otherwise, the transaction is still present. Note that skipping a transaction may cause data loss in the target database.

> **Note**
>
> To use `SKIPTRANS`, the specified transaction must be the oldest one in the list of transactions shown with `SHOWTRANS`. You can repeat the command for other transactions in order of their age.

transaction_ID
The transaction ID number. Get the ID number with `SHOWTRANS` or from an Extract runtime message.

FORCE
Valid for Oracle and SQL Server. Not valid for MySQL.
Use `FORCE` to bypass the prompt that confirms your intent to skip the transaction.

STATUS
Returns a detailed status of the processing state, including current position and activity. Possible processing status messages on the `Current status` line are:

- `Delaying` – waiting for more data
- `Suspended` – waiting to be resumed
- `Processing data` – processing data
- `Starting initial load` – starting an initial load task
- `Processing source tables` – processing data for initial load task
- `Reading from data source` – reading from the data source, such as a source table or transaction log
- `Adding record to transaction list` – adding a record to the file memory transaction list
- `At EOF (end of file)` – no more records to process

In addition to the preceding statuses, the following status notations appear during an Extract recovery after an abend event. You can follow the progress as Extract continually changes its log read position over the course of the recovery.

- `In recovery[1]` – Extract is recovering to its checkpoint in the transaction log.
- `In recovery[2]` – Extract is recovering from its checkpoint to the end of the trail.
- `Recovery complete` – The recovery is finished, and normal processing will resume.

STOP
Stops Extract. If there are any long-running transactions (based on the WARNLONGTRANS parameter), the following message will be displayed:

```
Sending STOP request to EXTRACT EXTE...
There are open, long-running transactions. Before you stop Extract, make the
archives containing data for those transactions available for when Extract
restarts. To force Extract to stop, use the SEND EXTRACT group, FORCESTOP
command.
Oldest redo log file necessary to restart Extract is:
Redo Thread 1, Redo Log Sequence Number 150, SCN 31248005, RBA 2912272.
```

TRACE[2] {*file_name* | OFF}
Turns tracing on and off. Tracing captures information to the specified file to reveal processing bottlenecks. Contact Oracle Support for assistance if the trace reveals significant processing bottlenecks.

TRACE
Captures step-by-step processing information.

TRACE2
Identifies code segments rather than specific steps.

file_name
Specifies the name of the file to which the trace information is written. If a trace is already running when SEND EXTRACT is issued with TRACE, the existing trace file is closed and the trace is resumed to the new file specified with *file_name*.

OFF
Turns off tracing.

TRACE OFF *file_name*
Turns tracing off only for the specified trace file.

TRACEINIT
Resets tracing statistics back to 0 and then starts accumulating statistics again. Use this option to track the current behavior of processing, as opposed to historical.

TRANLOGOPTIONS INTEGRATEDPARAMS(*parameter_specification*)
(Oracle) Supports an integrated Extract. Sends a parameter specification to the database inbound server while Extract is running in integrated mode. Only one parameter specification can be sent at a time with this command. You can send multiple parameter changes, issue multiple SEND EXTRACT commands.
To preserve the continuity of processing, the parameter change is made at a transaction boundary.

TRANLOGOPTIONS {PURGEORPHANEDTRANSACTIONS | NOPURGEORPHANEDTRANSACTIONS}
Valid for Oracle RAC. Enables or disables purging of orphaned transactions that occur when a node fails and Extract cannot capture the rollback.

TRANLOGOPTIONS TRANSCLEANUPFREQUENCY *minutes*
Valid for Oracle RAC. Specifies the interval, in minutes, after which Oracle GoldenGate scans for orphaned transactions and then re-scans to confirm and delete them. Valid values are from 1 to 43200 minutes. Default is 10 minutes.

VAMMESSAGE '*Teradata_command*'
VAMMESSAGE { 'ARSTATS' | 'INCLUDELIST [*filter*]' | 'EXCLUDELIST [*filter*]' }
Sends a command to the capture API that is used by Extract.
A Teradata command can be any of the following:

'control:terminate'
Stops a replication group. Required before dropping or altering a replication group in Teradata.

'control:suspend'
Suspends a replication group. Can be used when upgrading Oracle GoldenGate.

'control:resume'
Resumes a replication group after it has been suspended.

'control:copy *database.table*'
Copies a table from the source database to the target database.

'ARSTATS'
Displays TMF audit reading statistics.

'FILELIST [*filter*]'
Displays the list of tables for which Extract has encountered data records in the audit trail that match the selection criteria in the `TABLE` parameters. The *`filter`* option allows use of a wildcard pattern to filter the list of tables returned. `GETFILELIST` can also be used in the same manner.

'EXCLUDELIST [*filter*]'
Displays the list of tables for which Extract has encountered data records in the audit trail that do not match the selection criteria in the `TABLE` parameters. The *`filter`* option allows use of a wildcard pattern to filter the list of tables returned. Certain system tables that are implicitly excluded will always be present in the list of excluded tables.

A SQL Server command can be the following:

VAMMESSAGE 'OPENTRANS'
Prints a list of open transactions with their transaction ID, start time, first LSN, and the number of operations they contain.

Examples

```
SEND EXTRACT exte, ROLLOVER
```

```
SEND EXTRACT exte, STOP
```

```
SEND EXTRACT exte, VAMMESSAGE 'control:suspend'
```

```
SEND EXTRACT exte, TRANLOGOPTIONS TRANSCLEANUPFREQUENCY 20
```

This example explains SKIPTRANS. Start with the following SHOWCH output, which shows that thread 2 is at Read Checkpoint #3.

```
INFO exte SHOWCH
Read Checkpoint #3
Oracle RAC Redo Log
Startup Checkpoint (starting position in the data source):
Thread #: 2
Sequence #: 17560
RBA: 65070096
Timestamp: 2011-07-30 20:04:47.000000
SCN: 1461.3499051750 (6278446271206)
Redo File: RAC4REDO/sss11g/onlinelog/group_4.292.716481937
```

Therefore, SKIPTRANS should be: SKIPTRANS *xid* THREAD 3.

```
SEND EXTRACT exte, SHOWTRANS COUNT 2
```

The following shows the default output of SHOWTRANS.

```
Oldest redo log file necessary to restart Extract is:
Redo Thread 1, Redo Log Sequence Number 148, SCN 30816254, RBA 17319664
------------------------------------------------------------
XID                : 5.15.52582
Items              : 30000
Extract            : JC108XT
Redo Thread        : 1
Start Time         : 2011-01-18:12:51:27
SCN                : 20634955
Redo Seq           : 103
Redo RBA           : 18616848
Status             : Running
------------------------------------------------------------
XID                : 7.14.48657
Items              : 30000
Extract            : JC108XT
Redo Thread        : 1
Start Time         : 2011-01-18:12:52:14
SCN                : 20635145
Redo Seq           : 103
Redo RBA           : 26499088
Status             : Running
```

The following example shows SHOWTRANS output with TABULAR in effect (view is truncated on right).

```
XID         Items  Extract   Redo Thread  Start Time
5.15.52582  30000  JC108XT       1           2011-01-18:12:52:14

Dumping transaction memory at 2011-01-21 13:36:54.
Record #1:
Header (140 bytes):
```

```
       0: 0000 0A4A 0000 FFFF 0000 0000 0057 6C10        ...J.........Wl.
      16: 02FF 3F50 FF38 7C40 0303 4141 414E 5A77        ..?P.8|@..AAANZw
      32: 4141 4641 4141 4B6F 4941 4144 0041 4141        AAFAAAKoIAAD.AAA
      48: 4E5A 7741 4146 4141 414B 6F49 4141 4400        NZwAAFAAAKoIAAD.
      64: 4141 414E 5A77 414A 2F41 4142 7A31 7741        AAANZwAJ/AABz1wA
      80: 4141 0041 4141 4141 4141 4141 4141 4141        AA.AAAAAAAAAAAA
      96: 4141 4141 4100 0000 0140 FF08 0003 0000        AAAAA....@......
     112: 0000 0000 0000 70FF 0108 FFFF 0001 4A53        ......p.......JS
     128: 554E 2E54 4355 5354 4D45 5200                  UN.TCUSTMER.

Data (93 bytes):
       0: 2C00 0400 0400 0000 0100 0200 0300 0000        ,...............
      16: 0000 0000 0800 0000 1800 0000 2000 0400        ............ ...
      32: 1000 0600 0200 0000 284A 414E 456C 6C6F        ........(JANEllo
      48: 6352 4F43 4B59 2046 4C59 4552 2049 4E43        cROCKY FLYER INC
      64: 2E44 454E 5645 5220 6E43 4F20 7365 7400        .DENVER nCO set.
      80: 0000 0000 0000 0C00 0000 0000 00              ..............
```

When analyzing the summary output of `SHOWTRANS`, understand that it shows all currently running transactions on the database (as many as will fit into a predefined buffer). Extract must track every open transaction, not just those that contain operations on tables configured for Oracle GoldenGate.

The `Items` field of the `SHOWTRANS` output shows the number of operations in the transaction that have been captured by Oracle GoldenGate so far, not the total number of operations in the transaction. If none of the operations are for configured tables, or if only some of them are, then `Items` could be 0 or any value less than the total number of operations.

The `Start Time` field shows the timestamp of the first operation that Oracle GoldenGate extracts from a transaction, not the actual start time of the transaction itself.

> **Note**
>
> Command output may vary somewhat from the examples shown due ongoing enhancements of Oracle GoldenGate.

The following example shows sending BR request to Extract `exte`

```
SEND exte BR BRSTATUS
```

Output displays the following:

```
Bounded Recovery Parameter:
Options    = _BRDEBUG _NOBRCLEANUP _BRFORCE_ASSERT
BRINTERVAL = 40SECONDS
BRDIR      = /home/mpopeang/ogg_test/
Bounded Recovery Status: IN PROGRESS
Checkpoint interval = 40SECONDS
Next checkpoint  = 2020-01-15 21:10:47
Last checkpoint# = 49
Last checkpoint  = 2020-01-15 21:10:07
Total objects    = 65
```

```
Total size       = 426 MB
Outstanding objects = 58
Outstanding size    = 384 MB
Object pool 1: p12733_extr:RECOVERY: COMPLETE: start:SeqNo: 1580,
RBA: 793460, SCN: 0.664178312(664178312), Timestamp: 2020-01-15
20:35:45.000000,
Thread: 1, end=SeqNo:1580,
RBA: 793460, SCN: 0.664178312 (664178312), Timestamp: 2020-01-15
20:35:45.000000,
Thread: 1, complete=SeqNo: 1580, RBA: 793460,
SCN:0.664178312 (664178312), Timestamp: 2020-01-15 20:35:45.000000, Thread: 1
at 2020-01-15 20:38:52.435830
CHECKPOINT: start=SeqNo: 1637, RBA: 10182312, SCN: 0.669567539 (669567539),
Timestamp: 2020-01-15 21:09:59.000000, Thread: 1, end=SeqNo: 1637,
RBA:10182312, SCN: 0.669567539 (669567539), Timestamp: 2020-01-15
21:09:59.000000,
Thread: 1
```

The following example provides details on each of the object pools persisted by BR:

```
SEND exte BR BRSTATS
```

Output:

```
Object pool #0, instance: 1, id: p12733_extr
```

Object sizes through life of Extract group:

```
Sizes in bytes            :         POs
  512K      to        1M-1         :         13
    1M      to        2M-1         :         37
    4M      to        8M-1         :        398
    8M      to       16M-1         :          2
   16M      to       32M-1         :         10
```

Object ages through life of Extract group:

```
duration: BCP intervals in the life of the PO
  duration                         0:        230
  duration         30 to          39:        230
```

Extant object sizes as of most recent BCP:

```
Sizes in bytes                   :        POs
          512K      to        1M-1        :          8
            1M      to        2M-1        :         17
            4M      to        8M-1        :        200
           16M      to       32M-1        :          5
```

Extant object ages as of most recent BCP:

```
duration: BCP intervals in the life of the PO
  duration         30 to            39:        230
```

Examples for using `CACHEMGR` options
Returns all cache manager statistics.

```
SEND EXTRACT EXT1, CACHEMGR CACHESTATS ALL
```

Returns minimal cache manager statistics.

```
SEND EXTRACT EXT1, CACHEMGR CACHESTATS MINIMAL
```

Returns superpool cache manager statistics.

```
SEND EXTRACT EXT1, CACHEMGR CACHESTATS SUPERPOOL
```

Returns Cache Manager statistics for the current runtime only.

```
SEND EXTRACT EXT1, CACHEMGR CACHESTATS CURRENTONLY
```

Returns statistics for all object pools.

```
SEND EXTRACT EXT1, CACHEMGR CACHESTATS POOLS
```

Returns statistics for the specified object pool only. Replace *n* with the object pool number.

```
SEND EXTRACT EXT1, CACHEMGR CACHESTATS POOL n
```

Example: Returns statistics for object pool 1 only.

```
SEND EXTRACT EXT1, CACHEMGR CACHESTATS POOL 1
```

Returns statistics for queues.

```
SEND EXTRACT EXT1, CACHEMGR CACHESTATS QUEUES
```

Returns cache manager free queue statistics.

```
SEND EXTRACT EXT1, CACHEMGR CACHEQUEUES
```

Returns cache virtual memory usage statistics.

```
SEND EXTRACT EXT1, CACHEMGR CACHEVMUSAGE
```

SEND REPLICAT

Use `SEND REPLICAT` to communicate with a starting or running Replicat process. The request is processed as soon as Replicat is ready to accept commands from users.

Syntax

```
SEND REPLICAT group_name[threadID],
{
CACHEMGR {CACHESTATS | CACHEQUEUES | CACHEVMUSAGE} |
DEPENDENCYINFO [TXNCOUNT num] |
FORCESTOP |
GETLAG |
GETPARAMINFO [parameter_name] [FILE output_file] |
HANDLECOLLISIONS | NOHANDLECOLLISIONS [table_spec] |
INTEGRATEDPARAMS(parameter_specification) |
LOGEND |
REPORT [HANDLECOLLISIONS [table_spec]] |
RESUME |
STATUS |
STOP |
TRACE[2] [DDLINCLUDE | DDL[ONLY]] file_name |
TRACE[2] OFF |
TRACE OFF file_name |
TRACEINIT |
THREADS (threadID[, threadID][, ...][, thread_range[, thread_range][, ...])
}
```

***group_name*[*threadID*]**
The name of the Replicat group or the name of a specific thread of a coordinated Replicat, for example `fin003`. If the command is issued for a specific thread, then an option that is used applies only to that thread. As an alternative, you can issue `SEND REPLICAT` with the `THREADS` option instead of including *`threadID`* with the group name. If Replicat is not running, an error is returned.

CACHEMGR {CACHESTATS | CACHEQUEUES | CACHEVMUSAGE}
Returns statistics about the Oracle GoldenGate memory cache manager. `CACHEMGR` should only be used as explicitly directed by Oracle Support.

CACHESTATS
Returns statistics for file caching.

CACHEQUEUES
Returns statistics for the free queues only.

CACHEVMUSAGE
Returns statistics for the virtual memory usage.

DEPENDENCYINFO [TXNCOUNT num]
Prints out information from the PR transaction dependency graph. It first prints the transaction groups that are currently being executed and then the waiting transactions that are dependent on some other transactions. `TXNCOUNT` determines the number of waiting transactions to print. The default value is 10 and the maximum value is 99.

FORCESTOP
Forces Replicat to stop, bypassing any notifications. This command will roll back any active transaction and stop the process immediately. This command applies to Replicat as a whole and cannot be used for a specific Replicat thread.

GETLAG
Shows a true lag time between Replicat and the trail. Lag time is the difference, in seconds, between the time that the last record was processed by Replicat and the timestamp of the record in the trail. The results are the same as `LAG REPLICAT`.

GETPARAMINFO [*parameter_name*] [FILE *output_file*]
Use `GETPARAMINFO` to query runtime parameter values of a running instance, including Extract, Replicat, and Manager. You can query for a single parameter or all parameters and send the output to the console or a text file

parameter_name
The default behavior is to display all parameters in use, meaning those parameters that have ever been queried by the application, parameters, and their current values. If you specify a particular parameter, then the output is filtered by that name.

FILE *output_file*
The name of the text file that your output is redirected to.

HANDLECOLLISIONS | NOHANDLECOLLISIONS [*table_spec*]
Control `HANDLECOLLISIONS` behavior. Instead of using this option, you can specify the `HANDLECOLLISIONS` or `NOHANDLECOLLISIONS` parameter in the Replicat parameter file. See `HANDLECOLLISIONS | NOHANDLECOLLISIONS` in *Parameters and Functions Reference for Oracle GoldenGate*. This command can be sent directly to an individual thread by means of `SEND REPLICAT` *`group_name[threadID]`* or you can use the `THREADS` option to send the command through the coordinator thread to affect multiple threads.

HANDLECOLLISIONS
Use `HANDLECOLLISIONS` to enable automatic error handling when performing initial data loads while the source database is active. Make certain to disable `HANDLECOLLISIONS`, either by issuing `SEND REPLICAT` with the `NOHANDLECOLLISIONS` option or by removing the parameter from the parameter file, after the initial load is complete and online data changes have been applied to the target tables.

> **Note**
>
> The message returned by `SEND REPLICAT` with `HANDLECOLLISIONS`, when issued for a specific Replicat thread, shows that the command set `HANDLECOLLISIONS` for all `MAP` statements, not only the one handled by the specified thread. This is a known issue. The command actually affects only the `MAP` statement that includes the specified thread.

NOHANDLECOLLISIONS
Turns off the `HANDLECOLLISIONS` parameter but does not remove it from the parameter file. To avoid enabling `HANDLECOLLISIONS` the next time Replicat starts, remove it from the parameter file.

table_spec
`table_spec` restricts `HANDLECOLLISIONS` or `NOHANDLECOLLISIONS`to a specific target table or a group of target tables specified with a standard wildcard (*).

INTEGRATEDPARAMS(*parameter_specification*)
(Oracle) Supports an integrated Replicat. Sends a parameter specification to the database inbound server while Replicat is running in integrated mode. Only one parameter specification can be sent at a time with this command. To send multiple parameter changes, issue multiple `SEND REPLICAT` commands as in the following example.

```
SEND REPLICAT repe INTEGRATEDPARAMS (parallelism 4)
SEND REPLICAT repe INTEGRATEDPARAMS (max_sga_size 250)
```

To preserve the continuity of processing, the parameter change is made at a transaction boundary.

LOGEND
Confirms whether or not Replicat has processed all of the records in the data source.

REPORT [HANDLECOLLISIONS [*table_spec*]]
Generates an interim statistical report to the Extract report file. The statistics that are displayed depend upon the configuration of the `STATOPTIONS` parameter when used with the `RESETREPORTSTATS | NORESETREPORTSTATS` option. See `STATOPTIONS`.

HANDLECOLLISIONS
Shows tables for which `HANDLECOLLISIONS` has been enabled.

table spec
Restricts the output to a specific target table or a group of target tables specified with a standard wildcard (*).

RESUME
Resumes (makes active) a process that was suspended by an `EVENTACTIONS SUSPEND` event. The process resumes normal processing from the point at which it was suspended.

STATUS
Returns the current location within the trail and information regarding the current transaction. Fields output are:

- Processing status (per thread, if Replicat is coordinated)
- Position in the trail file (per thread, if Replicat is coordinated)
- Trail sequence number (per thread, if Replicat is coordinated)
- RBA in trail
- Trail name

Possible processing status messages are:

- `Delaying` – waiting for more data
- `Suspended` – waiting to be resumed
- `Waiting on deferred apply` – delaying processing based on the `DEFERAPPLYINTERVAL` parameter.
- `Processing data` – processing data
- `Skipping current transaction` – `START REPLICAT` with `SKIPTRANSACTION` was used.
- `Searching for START ATCSN` *csn* – `START REPLICAT` with `ATCSN` was used.
- `Searching for START AFTERCSN` *csn* – `START REPLICAT` with `AFTERCSN` was used.

- `Performing transaction timeout recovery` – Canceling current incomplete transaction and repositioning to start new one (see the `TRANSACTIONTIMEOUT` parameter).
- `Waiting for data at logical EOF after transaction timeout recovery` – Waiting to receive remainder of incomplete source transaction after a `TRANSACTIONTIMEOUT` termination.
- `At EOF (end of file)` – no more records to process

Possible thread status messages when `THREADS` is used or the command is issued for a specific thread are:

- `Waiting for consensus stop point`: This indicates that the threads are attempting to synchronize for a barrier transaction.
- `Waiting for Watermark`: Indicates that all threads are attempting to stop at the same transaction boundary in the trail, known as the global watermark.
- `Waiting on all threads to start up`: Indicates that the thread is waiting for all of the threads to start after a successful barrier transaction or a Replicat startup.

Possible coordinator thread status messages are:

- `Waiting for all threads to register`: Indicates that the `MAP` statements are all being parsed to determine the thread IDs that are specified in them.
- `Processing data`: Indicates that data is being processed normally.
- `Suspended, waiting to be resumed`: Indicates that a `SEND REPLICAT` command with a `SUSPEND` request was sent to Replicat.
- `At EOF`: Indicates that there is no more data in the trail to process.
- `Waiting to register MAP statistics`: Indicates that Replicat is collecting processing statistics to send to the report file.

STOP
Stops Replicat gracefully. This command applies to Replicat as a whole and cannot be used for a specific Replicat thread.

THREADS (*threadID*[, *threadID*][, ...][, *thread_range*[, *thread_range*][, ...])
Issues the command only for the specified thread or threads of a coordinated Replicat. You can use this option or you can use *groupname* with *threadID*. Without either of those options, the command applies to all active threads.

***threadID*[, *threadID*][, ...]**
Specifies a thread ID or a comma-delimited list of threads in the format of `threadID, threadID, threadID`.

***thread_range*[, *thread_range*][, ...]**
Specifies a range of threads in the form of `threadIDlow-threadIDhigh` or a comma-delimted list of ranges in the format of `threadIDlow-threadIDhigh, threadIDlow-threadIDhigh`.

A combination of these formats is permitted, such as threadID, threadID, `threadIDlow-threadIDhigh`.

TRACE[2] [DDLINCLUDE | DDL[ONLY]] *file_name*
Turns tracing on and off. Tracing captures information to the specified file to reveal processing bottlenecks. Tracing also can be enabled by means of the Replicat parameters `TRACE` and `TRACE2`.

If the Replicat is in coordinated mode and TRACE is used with a THREADS list or range, a trace file is created for each currently active thread. Each file name is appended with its associated thread ID. This method of identifying trace files by thread ID does not apply when SEND REPLICAT is issued by groupname with threadID (as in SEND REPLICAT fin003 TRACE...) or when only one thread is specified with THREADS.
Contact Oracle Support for assistance if the trace reveals significant processing bottlenecks.

TRACE
Captures step-by-step processing information.

TRACE2
Identifies code segments rather than specific steps.

DDLINCLUDE | DDLONLY
(Replicat only) Enables DDL tracing and specifies how DDL tracing is included in the trace report.

- DDLINCLUDE includes DDL tracing in addition to regular tracing of transactional data processing.
- DDL[ONLY] excludes the tracing of transactional data processing and only traces DDL. This option can be abbreviated to DDL.

file_name
file_name specifies the relative or fully qualified name of a file to which Oracle GoldenGate logs the trace information. If a trace is already in progress, the existing trace file is closed and the trace resumes to the file specified with *file_name*. For example:

```
SEND REPLICAT group_name TRACE file_name DDLINCLUDE
```

If no other options will follow the file name, the FILE keyword can be omitted, for example:

```
SEND REPLICAT group_name TRACE DDLINCLUDE file_name
```

TRACE[2] OFF
Turns off tracing.

TRACE OFF *file_name*
Turns tracing off only for the specified trace file. This option supports the EVENTACTIONS feature, where there can be multiple trace files due to multiple EVENTACTIONS statements.

TRACEINIT
Resets tracing statistics back to 0 and then starts accumulating statistics again. Use this option to track the current behavior of processing, as opposed to historical.

Examples

```
SEND REPLICAT repe, HANDLECOLLISIONS
```

```
SEND REPLICAT repe, REPORT HANDLECOLLISIONS rep_*
```

```
SEND REPLICAT repe, GETLAG


SEND REPLICAT repe, INTEGRATEDPARAMS(parallelism 10)
```

The following gets lag for thread 3 of a coordinated Replicat.

```
SEND REPLICAT repe, GETLAG
```

The following enables tracing for only thread 1 of a coordinated Replicat. In this case, because only one thread is being traced, the trace file will not have a *threadID* extension. The file name is `trace.trc`.

```
SEND REPLICAT repe, TRACE THREADS(1) FILE ./dirrpt/trace.trc
```

The following enables tracing for threads 1,2, and 3 of a coordinated Replicat. Assuming all threads are active, the tracing produces files `trace001`, `trace002`, and `trace003`.

```
SEND REPLICAT repe TRACE THREADS(1-3) FILE ./dirrpt/trace.trc
```

The following enables tracing only for thread 1 of a coordinated Replicat. Because the command was issued directly for thread 1 without the use of a `THREAD` clause, the trace file is named `trace` (without a thread ID suffix).

```
SEND REPLICAT repe TRACE FILE ./dirrpt/trace.trc
```

SET EDITOR

Use `SET EDITOR` to change the default text editor for the current session of Admin Client. The default editors are Notepad for Windows and vi for UNIX. CLI input, including to create parameter files, takes the character set of the local operating system.

Syntax

```
SET EDITOR program_name
```

program_name
Any text editor.

Example

The following example changes the default editor to Notepad++.

```
SET EDITOR notepad++
```

SET COLOR

This commands allows you to enable or disable colored text in the Admin Client.

Syntax

```
SET COLOR ( ON | OFF )
```

SET DEBUG

Use `SET DEBUG` to enable or disable debugging mode for the Admin Client. By default, this is set by the value of the environment variable, `ADMINCLIENT_DEBUG`. Use the `SHOW` command to see the value of the `SET DEBUG` variable.

Syntax

```
SET DEBUG ON | OFF
```

`ON`
Debugging mode is enabled.

`OFF`
Debugging mode is disabled.

SET INSTANTIATION CSN

Use `SET INSTANTIATION CSN` on your target database to set the instantiation CSN manually. This command requires `DBLOGIN`. It enables a Replicat with the `DBOPTIONS ENABLE_INSTANTIATION_FILTERING` option to filter out records below the specified CSN for any object without Oracle data pump import instantiation information. It is an alternative to specifying `@FILTER(@GETENV('TRANSACTION','CSN'))`.

To enable instantiation SCN filtering, you must do the following:

1. Your Replicat parameter file must contain `DBOPTIONS ENABLE_INSTANTIATION_FILTERING`.
2. The instantiation SCNs must be set at the target database for each table.

   You can do this using one of the following two methods:

   Automatically set the source SCN by the Oracle data pump upon import if the tables were prepared at the source database using `ADD TRANDATA PREPARECSN` or `ADD SCHEMATRANDATA PREPARECSN` prior to the Oracle data pump export.

   or

   Manually set the instantiation source SCN at the target database using this command.

Syntax

```
SET INSTANTIATION CSN csn FOR [schema.]table FROM source-database-name
```

`csn`
The CSN number that instantiation will begin.

`[schema.]table`
The name of the table to set the instantiation CSN on. If no schema is provided, the `DBLOGIN` user is used.

source-database-name
The global name of the source database for which this is a target.

Example

```
SET INSTANTIATION CSN 12345678 FOR hr.employees FROM orcl.com
```

SET PAGER

Use `SET PAGER` to set the default text viewer program for viewing parameter and report files. By default, this is set by the value of the environment variable, `PAGER`, On UNIX and Linux it defaults to `less` and to `more` on Windows.

Use the `SHOW` command to see the value of the `SET PAGER` variable.

Syntax

```
SET PAGER command
```

`command`

Any text viewer.

Example

```
SET PAGER wordpad
```

SET VERBOSE

This command allows you to enable or disable verbose command result output.

Syntax

```
SET VERBOSE (ON | OFF)
```

SHELL

Use `SHELL` to execute shell commands from within the CLI.

In Admin Client, this command is run on the local system and not on the system where the Administration Service or Service Manager is running.

Syntax

```
SHELL command
```

command
The system command to execute.

Examples

```
SHELL dir prod\*
```

SHOW

Use SHOW to display the Oracle GoldenGate environment.

Syntax

```
SHOW
```

Example

The following are samples of SHOW output. Additional entries may be displayed, depending on the database type.

```
Current directory: /scratch/ogg/sa/bin
DEBUG        : OFF
EDITOR        : vi
PAGER        : more
```

or

```
Parameter settings:
SET DEBUG      OFF
Current directory: C:\GG_81
Using subdirectories for all process files
Editor:  notepad
Reports (.rpt)                C:\GG_81\dirrpt
Parameters (.prm)             C:\GG_81\dirprm
Replicat Checkpoints (.cpr)   C:\GG_81\dirchk
Extract Checkpoints (.cpe)    C:\GG_81\dirchk
Process Status (.pcs)         C:\GG_81\dirpcs
SQL Scripts (.sql)            C:\GG_81\dirsql
Database Definitions (.def)   C:\GG_81\dirdef
```

START DEPLOYMENT

Use START DEPLOYMENT to start the specified deployment.

Syntax

```
START DEPLOYMENT deployment-name-wildcard
```

deployment-name-wildcard
The name of the deployment or a wildcard (*) to specify multiple deployments. For example, T* sends the command to all deployments whose group names begin with T.

Example

```
START DEPLOYMENT NORTH
```

START DISTPATH

Use `START DISTPATH` to start a distribution path. To confirm that the distribution path has started, use the `INFO DISTPATH` command. To change the distribution path start point, use the `ALTER DISTPATH` command.

Syntax

```
START DISTPATH path-name
         [atCSN csn_value | afterCSN csn_value]
```

path-name
The name of the distribution path.

ATCSN
Directs the distribution path to position its start point at the first transaction that has the specified CSN. Any transactions in the data source that have CSN values less than the specified one are skipped.

AFTERCSN
Directs the distribution path to position its start point at the beginning of the first transaction after the one that has the specified CSN. Any transactions in the data source that have CSN values that are less than, or equal to, the specified one are skipped.

START ER

Use the `START ER` to start multiple Extract and Replicat groups as a unit. Use it with wildcards to affect every Extract and Replicat group that satisfies the wildcard. For descriptions and optional parameters for this command, see `INFO EXTRACT`.

Syntax

```
START  ER group_name
```

group_name
The wildcard specification for the groups that you want to affect with the command. Oracle GoldenGate automatically increases internal storage to track up to 100,000 wildcard entries.

Example

```
START ER *
```

START EXTRACT

Use `START EXTRACT` to start the Extract process. To confirm that Extract has started, use the `INFO EXTRACT` or `STATUS EXTRACT` command. Extract can be started at its normal start point

(from initial or current checkpoints) or from an alternate, user-specified position in the data source.

Normal Start Point

Without options, `START EXTRACT` directs an Extract to start processing at one of the following locations in the data source to maintain data integrity:

- After graceful or abnormal termination: At the first unprocessed transaction in the data source from the previous run, as represented by the current read checkpoint.
- First-time startup after the group was created: At the start point specified with the `ADD EXTRACT` command.

Alternate Start Point

Before starting Extract with `ATCSN` or `AFTERCSN`, you must establish a physical starting location with one of the following commands:

- `ADD EXTRACT` with the `BEGIN` option set to a timestamp that is earlier than the CSN value specified with `ATCSN` or `AFTERCSN`. The transaction log that contains the timestamp and every log thereafter must be available on the system before Extract is started.
- `ALTER EXTRACT` to the sequence number of the log that contains the CSN specified with `ATCSN` or `AFTERCSN`.

Syntax

```
START EXTRACT group_name
        [ATCSN csn | AFTERCSN csn]
        [BRoptions]
```

group_name
The name of an Extract group or a wildcard (*) to specify multiple groups. For example, `T*` starts all Extract groups whose names begin with T.

ATCSN *csn* | AFTERCSN *csn*
Specifies an alternate start point.

ATCSN
Directs Extract to position its start point at the first transaction that has the specified CSN. Any transactions in the data source that have CSN values less than the specified one are skipped.

AFTERCSN
Directs Extract to position its start point at the beginning of the first transaction after the one that has the specified CSN. Any transactions in the data source that have CSN values that are less than, or equal to, the specified one are skipped.

csn
Specifies a CSN value. Enter the CSN value in the format that is valid for the database. Extract abends if the format is invalid and writes a message to the report file. To determine the CSN to supply after an initial load is complete, use the serial identifier at which the load utility completed. Otherwise, follow the instructions in the initial load procedure for determining when to start Extract.

The following are additional guidelines to observe when using `ATCSN` and `AFTERCSN`:

- The CSN is stored in the file header so that it is available to downstream processes.
- When a record that is specified with a CSN is found, Extract issues a checkpoint. The checkpoint ensures that subsequent Extract startups begin from the requested location, and not from a point prior to the requested CSN.
- You must establish a physical start point in the transaction log or trail for Extract with `ADD EXTRACT` or `ALTER EXTRACT` before using `ATCSN` or `AFTERCSN`. These options are intended to be an additional filter after Extract is positioned to a physical location in the data source.

BR*options*
Extract can be started with BR options. Here are some examples of the BR options:

- `START EXTRACT BROFF`
- `START EXTRACT BRRESET`
- `START EXTRACT BRInterval # BRKEEPSTALEFILES`
- `START EXTRACT BRKEEPSTALEFILES`
- `START EXTRACT BRFSOPTION`

See `BR` in *Parameters and Functions Reference for Oracle GoldenGate* for details.

Examples

```
START EXTRACT exte
```

```
START EXTRACT exte ATCSN 684993
```

```
START EXTRACT exte AFTERCSN 684993
```

START RECVPATH

Use `START RECVPATH` to start a target-initiated distribution path in the Receiver Service. To confirm that the distribution path has started, use the `INFO RECVPATH` command. To change the distribution path start point, use the `ALTER RECVPATH` command.

Syntax

```
START RECVPATH path-name
```

path-name
The name of the distribution path.

START REPLICAT

Use `START REPLICAT` to start Replicat. To confirm that Replicat has started, use the `INFO REPLICAT` or `STATUS REPLICAT` command.

When starting an integrated Replicat group for an Oracle target database, `START REPLICAT` automatically registers Replicat with the target database.

A coordinated Replicat can only be started as a whole. There is no option to start individual threads. If the prior shutdown of a coordinated Replicat was not clean, the threads may have stopped at different positions in the trail file. If this happens, `START REPLICAT` writes a warning if the parameter file was changed since the prior run and raises an error if the number of threads was changed.

Normal Start Point

Replicat can be started at its normal start point (from initial or current checkpoints) or from an alternate, user-specified position in the trail.

`START REPLICAT`, without any options, causes Replicat to start processing at one of the following points to maintain data integrity:

- After graceful or abnormal termination: At the first unprocessed transaction in the trail from the previous run, as represented by the current read checkpoint.
- First-time startup after the group was created: From the beginning of the active trail file (`seqno 0`, `rba 0`).

Alternate Start Point

The `SKIPTRANSACTION`, `ATCSN`, and `AFTERCSN` options of `START REPLICAT` cause Replicat as a whole, or specific threads of a coordinated Replicat, to begin processing at a transaction in the trail other than the normal start point. Use these options to:

- Specify a logical recovery position when an error prevents Replicat from moving forward in the trail. Replicat can be positioned to skip the offending transaction or transactions, with the understanding that the data will not be applied to the target.
- Skip replicated transactions that will cause duplicate-record and missing-record errors after a backup is applied to the target during an initial load. These options cause Replicat to discard transactions that occurred earlier than the most recent set of changes that were captured in the backup. You can map the value of the serial identifier that corresponds to the completion of the backup to a CSN value, and then start Replicat to begin applying transactions from the specified CSN onward.

> **Note**
>
> Skipping a transaction, or starting at or after a CSN, might cause Replicat to start more slowly than normal, depending on how much data in the trail must be read before arriving at the appropriate transaction record. To view the startup progress, use the `SEND REPLICAT` command with the `STATUS` option. To omit the need for Replicat to read through transactions that ultimately will be skipped, you can use the `ATCSN` or `AFTERCSN` option when starting Extract, so that those transactions are omitted from the trail, see START EXTRACT.

Syntax

```
START REPLICAT group_name
[SKIPTRANSACTION | {ATCSN csn | AFTERCSN csn}]
[FILTERDUPTRANSACTIONS | NOFILTERDUPTRANSACTIONS]
[THREADS (threadID[, threadID][, ...][, thread_range[, thread_range][, ...])
```

group_name
The name of a Replicat group or a wildcard (*) to specify multiple groups. For example, `T*` starts all Replicat groups whose names begin with T.

SKIPTRANSACTION
Causes Replicat to skip the first transaction that it encounters after its expected startup position in the trail. All operations from that first transaction are excluded.
If the `MAXTRANSOPS` parameter is also being used for this Replicat, it is possible that the process will start to read the trail file from somewhere in the middle of a transaction. In that case, the remainder of the partial transaction is skipped, and Replicat resumes normal processing from the next begin-transaction record in the file. The skipped records are written to the discard file if the `DISCARDFILE` parameter is being used; otherwise, a message is written to the report file that is similar to:

```
User requested START SKIPTRANSACTION. The current transaction will be
skipped. Transaction ID txid, position Seqno seqno, RBA rba
```

`SKIPTRANSACTION` is valid only when the trail that Replicat is reading is part of an online change synchronization configuration (with checkpoints). Not valid for task-type initial loads (where `SPECIALRUN` is used with `ADD REPLICAT`).

ATCSN *csn* | AFTERCSN *csn*
Sets a user-defined start point at a specific CSN. When `ATCSN` or `AFTERCSN` is used, a message similar to one of the following is written to the report file:

```
User requested start at commit sequence number (CSN) csn-string
```

```
User requested start after commit sequence number (CSN) csn-string
```

General information about these options:

- Valid only when the trail that Replicat is reading is part of an online change synchronization configuration (with checkpoints). Not valid for task-type initial loads (where `SPECIALRUN` is used with `ADD REPLICAT`).
- To support starting at, or after, a CSN, the CSN is stored in the first trail record of each transaction. If Replicat is started with `AFTERCSN` against an earlier trail version, Replicat will abend and write an error to the report stating that the trail format is not supported.

  ATCSN
  Causes Replicat to start processing at the transaction that has the specified CSN. Any transactions in the trail that have CSN values that are less than the specified one are skipped.

  AFTERCSN
  Causes Replicat to start processing at the transaction that occurred after the one with the specified CSN. Any transactions in the trail that have CSN values that are less than, or equal to, the specified one are skipped.

csn

Specifies a CSN value. Enter the CSN value in the format that is valid for the database. See Commit Sequence Number (CSN) for CSN formats and descriptions. Replicat abends if the format is invalid and writes a message to the report file. To determine the CSN to supply after an initial load is complete, use the commit identifier at which the load utility completed the load. Otherwise, follow the instructions in the initial load procedure for determining when to start Replicat.

FILTERDUPTRANSACTIONS | NOFILTERDUPTRANSACTIONS
Causes Replicat to ignore transactions that it has already processed. Use when Extract is repositioned to a new start point (see the ATCSN or AFTERCSN option of "START EXTRACT") and you are confident that there are duplicate transactions in the trail that could cause Replicat to abend. This option requires the use of a checkpoint table. The default is FILTERDUPTRANSACTIONS. However, if you use NOFILTERDUPTRANSACTIONS, the integrated Replicat default setting is overridden and causes it to not filter the duplicates. So it has the same effect on both classic and integrated Replicat.

THREADS (*threadID*[, *threadID*][, ...][, *thread_range*[, *thread_range*][, ...])
Valid for SKIPTRANSACTION, ATCSN, and AFTERCSN when Replicat is in coordinated mode. Not valid for START REPLICAT without those options. Starts the specified Replicat thread or threads at the specified location.

***threadID*[, *threadID*][, ...]**
Specifies a thread ID or a comma-delimited list of threads in the format of threadID, threadID, threadID.

***thread_range*[, *thread_range*][, ...]**
Specifies a range of threads in the form of threadIDlow-threadIDhigh or a comma-delimited list of ranges in the format of threadIDlow-threadIDhigh, threadIDlow-threadIDhigh.

A combination of these formats is permitted, such as threadID, threadID, threadIDlow-threadIDhigh.

Examples

```
START REPLICAT reps
```

The following starts Replicat at an Oracle-specific CSN.

```
START REPLICAT reps, ATCSN 6488359
```

The following starts Replicat at a SQL Server-specific CSN after the one with the specified CSN.

```
START REPLICAT reps, AFTERCSN 0X000004D2:0000162E:0009
```

The following causes threads 4 and 5 of a coordinated Replicat to skip the first transaction after their last checkpoint when Replicat is started. If this were a 10-thread coordinated

Replicat, threads 0-3 and 6-10 would all start at the normal start point, that of their last checkpoint.

```
START REPLICAT reps SKIPTRANSACTION THREADS(4-5)
```

The following example causes threads 1-3 of a coordinated Replicat to start at CSN 6488359.

```
START REPLICAT reps ATCSN 6488359 THREADS(1-3)
```

START SERVICE

Use `START SERVICE` to start the specified Oracle GoldenGate services.

Syntax

```
START SERVICE service-name-wildcard
```

service-name-wildcard
The name of an service or a wildcard (*) to specify multiple services. Valid services are `ADMINSRVR`, `DISTSRVR`, `RECVSRVR`, and `PMSRVR`.

Example

```
START SERVICE ADMIN*
```

STATS DISTPATH

Use the `STATS DISTPATH` command to get the statistics for a distribution paths.

Syntax

```
STATS DISTPATH path-name
```

path-name
The name of the distribution path.

STATS ER

Use the `STATS ER` to get statistics on multiple Extract and Replicat groups as a unit. Use it with wildcards to affect every Extract and Replicat group that satisfies the wildcard. For descriptions and optional parameters for this command, see `INFO EXTRACT`.

Syntax

```
STATS  ER group_name
```

group_name
The wildcard specification for the groups that you want to affect with the command. For example, `T*` starts all groups whose names begin with T. Oracle GoldenGate automatically increases internal storage to track up to 100,000 wildcard entries.

Example

```
STATS  ER *T*
```

STATS EXTRACT

Use `STATS EXTRACT` to display statistics for one or more Extract groups. The output includes DML and DDL operations that are included in the Oracle GoldenGate configuration.

To get the most accurate number of operations per second that are being processed, do the following.

1. Issue the `STATS EXTRACT` command with the `RESET` option.
2. Issue the `STATS EXTRACT REPORTRATE` command. The `LATEST STATISTICS` field shows the operations per second.

> **Note**
>
> The actual number of DML operations executed on a DB2 database might not match the number of extracted DML operations reported by Oracle GoldenGate. DB2 does not log update statements if they do not physically change a row, so Oracle GoldenGate cannot detect them or include them in statistics.

> **Note**
>
> To get accurate statistics on a Teradata source system where Oracle GoldenGate is configured in maximum protection mode, issue `STATS EXTRACT` to the VAM-sort Extract, not the primary Extract. The primary Extract may contain statistics for uncommitted transactions that could be rolled back; whereas the VAM-sort Extract reports statistics only for committed transactions.

Syntax

```
STATS EXTRACT group_name
[, statistic]
[, DDLSONLY]
[, TABLE [container. | catalog.]schema.table]
[, TOTALSONLY [container. | catalog.]schema.table]
[, REPORTCDR]
[, REPORTCHARCONV]
[, REPORTFETCH | NOREPORTFETCH]
[, REPORTRATE time_units]
```

group_name
The name of an Extract group or a wildcard (*) to specify multiple groups. For example, `T*` returns statistics for all Extract groups whose names start with T.

statistic
The statistic to be displayed. More than one statistic can be specified by separating each with a comma, for example `STATS EXTRACT finance, TOTAL, DAILY`.

TOTAL
Displays totals since process startup.

DAILY
Displays totals since the start of the current day.

HOURLY
Displays totals since the start of the current hour.

LATEST
Displays totals since the last `RESET` command.

RESET
Resets the counters in the `LATEST` statistical field.

DDLSONLY
Displays the statistics for DDL statements including number of DDL statements in a readable format.

TABLE [*container*. | *catalog*.]*schema*.*table_name*
Displays statistics only for the specified table or a group of tables specified with a wildcard (*). The table name or wildcard specification must be fully qualified with the two-part or three-part name, for example `hr.emp` or `*.*.*`.

TOTALSONLY [*container*. | *catalog*.]*schema*.*table_name*
Summarizes the statistics for the specified table or a group of tables specified with a wildcard (*). The table name or wildcard specification must be fully qualified with the two-part or three-part name, for example `hr.emp` or `*.*.*`.

REPORTCDR
Shows statistics for Conflict Detection and Resolution. Statistics include:

- Total CDR conflicts
- CDR resolutions succeeded
- CDR resolutions failed
- CDR `INSERTROWEXISTS` conflicts
- CDR `UPDATEROWEXISTS` conflicts
- CDR `DELROWEXISTS` conflicts
- CDR `DELROWMISSING` conflicts

REPORTCHARCONV
Use only when `TABLE` parameters have a `TARGET` clause and character-set conversion is performed. The following statistics are added to the `STATS` output:
`Total column character set conversion failure`: the number of validation or conversion failures in the current Extract run.

`Total column data truncation`: the number of times that column data was truncated in the current Extract run as the result of character set conversion

`REPORTFETCH | NOREPORTFETCH`
Controls whether or not statistics about fetch operations are included in the output. The default is `NOREPORTFETCH`. The output of `REPORTFETCH` is as follows:

- `row fetch attempts`: The number of times Extract attempted to fetch a column value from the database when it could not obtain the value from the transaction log.
- `fetch failed`: The number of `row fetch attempts` that failed.
- row fetch by key: Valid for Oracle. The number of row fetch attempts that were made by using the primary key. The default is to fetch by row ID.

`REPORTRATE` *`time_units`*
Displays statistics in terms of processing rate rather than absolute values.

```
HR
MIN
SEC
```

Example

The following example displays total and hourly statistics per minute for a specific table, and it also resets the latest statistics and outputs fetch statistics.

```
STATS EXTRACT exte, TOTAL, HOURLY, TABLE hr.emp, REPORTRATE MIN, RESET,
REPORTFETCH


STATS EXTRACT exte, LATEST, REPORTFETCH
```

STATS RECVPATH

Use the `STATS RECVPATH` command to get the statistics for a target-initiated distribution path in the Receiver Server.

Syntax

```
STATS RECVPATH path-name
```

`path-name`
The name of the target-initiated distribution path.

STATS REPLICAT

Use `STATS REPLICAT` to display statistics for one or more Replicat groups. Thread statistics for a coordinated Replicat group are provided as follows.

Thread Lag Gap
The difference between the maximum lag and the minimum lag among all threads.

Coordinated Total DDLs
The total number of coordinated DDL transactions.

Coordinated Total PK-Update Transactions
The total number of coordinated transactions that involved an update to a primary key.

Coordinated Total EMI Transactions
The total number of coordinated `EVENTACTIONS` events.

Total Transactions with User-Requested Coordination
The total number of coordination's that were explicitly requested in the configuration by means of the `COORDINATED` option of the `MAP` parameter.

Average Coordination Time
The average time (in seconds) spent in coordination among all threads.

Syntax

```
STATS REPLICAT group_name
[, statistic]
[, DDLSONLY]
[, TABLE [container. | catalog.]schema.table]
[, TOTALSONLY [container. | catalog.]schema.table]
[, REPORTCDR]
[, REPORTCHARCONV]
[, REPORTDETAIL | NOREPORTDETAIL]
[, REPORTRATE {HR | MIN | SEC}]
```

group_name
The name of a Replicat group or a wildcard (*) to specify multiple groups. For example, `T*` shows statistics for all Replicat groups whose names begin with T.

statistic
The statistic to be displayed. More than one statistic can be specified by separating each with a comma, for example `STATS REPLICAT finance, TOTAL, DAILY.`

TOTAL
Displays totals since process startup.

DAILY
Displays totals since the start of the current day.

HOURLY
Displays totals since the start of the current hour.

LATEST
Displays totals since the last `RESET` command.

RESET
Resets the counters in the `LATEST` statistical field.

DDLSONLY
Displays the statistics for DDL statements including number of DDL statements in a readable format.

TABLE [*container*. | *catalog*.]*schema*.*table_name*]
Displays statistics only for the specified table or a group of tables specified with a wildcard (*). The table name or wildcard specification must be fully qualified with the two-part or three-part name, for example `hr.emp` or `*.*.*`.

TOTALSONLY [*container*. | *catalog*.]*schema*.*table_name*]
Summarizes the statistics for the specified table or a group of tables specified with a wildcard (*). The table name or wildcard specification must be fully qualified with the two-part or three-part name, for example `hr.emp` or `*.*.*`.

REPORTCDR
Shows statistics for Conflict Detection and Resolution. Statistics include:

- Total CDR conflicts
- CDR resolutions succeeded
- CDR resolutions failed
- CDR `INSERTROWEXISTS` conflicts
- CDR `UPDATEROWEXISTS` conflicts
- CDR `UPDATEROWMISSING` conflicts
- CDR `DELETEROWEXISTS` conflicts
- CDR `DELETEROWMISSING` conflicts

REPORTCHARCONV
Reports statistics for character validation when character-set conversion is performed. The following statistics are added to the `STATS` output:
`Total column character set conversion failure`: the number of validation or conversion failures in the current Replicat run.
`Total column data truncation`: the number of times that column data was truncated in the current Replicat run as the result of character set conversion

REPORTDETAIL | NOREPORTDETAIL
Controls whether or not the output includes operations that were not replicated as the result of collision errors. These operations are reported in the regular statistics (inserts, updates, and deletes performed) plus as statistics in the detail display, if enabled. For example, if 10 records were insert operations and they were all ignored due to duplicate keys, the report would indicate that there were 10 inserts and also 10 discards due to collisions. The default is `REPORTDETAIL`.

REPORTRATE {HR | MIN | SEC}
Displays statistics in terms of processing rate rather than absolute values.

HR
Sets the processing rate in terms of hours.

MIN
Sets the processing rate in terms of minutes.

SEC
Sets the processing rate in terms of seconds.

Examples

The following example displays total and hourly statistics per minute for a specific table, and it also resets the latest statistics. Statistics for discarded operations are not reported.

```
STATS REPLICAT finance, TOTAL, HOURLY, TABLE sales.acct, REPORTRATE MIN,
RESET, NOREPORTDETAIL
```

The following example displays the same statistics as the previous example, but for thread 3 of a coordinated Replicat group.

```
STATS REPLICAT fin003, TOTAL, HOURLY, TABLE sales.acct, REPORTRATE MIN,
RESET, NOREPORTDETAIL
```

STATUS DEPLOYMENT

Use `STATUS DEPLOYMENT` to see the status of the specified deployment.

Syntax

```
STATUS DEPLOYMENT deployment-name-wildcard
```

`deployment-name-wildcard`
The name of the deployment or a wildcard (*) to specify multiple deployments. For example, `T*` sends the command to all deployments whose group names begin with T.

Example

```
STATUS DEPLOYMENT production
```

STATUS ER

Use the `STATUS ER` to check the status of multiple Extract and Replicat groups as a unit. Use it with wildcards to affect every Extract and Replicat group that satisfies the wildcard. For descriptions and optional parameters for this command, see `STATUS EXTRACT`.

Syntax

```
STATUS  ER group_name
```

`group_name`
The wildcard specification for the groups that you want to affect with the command. For example, `T*` shows statistics for all groups whose names begin with T. Oracle GoldenGate automatically increases internal storage to track up to 100,000 wildcard entries.

Example

```
STATUS  ER *
```

STATUS EXTRACT

Use `STATUS EXTRACT` to determine whether or not Extract is running. A status of `RUNNING` can mean one of the following:

- Active: Running and processing (or able to process) data. This is the normal state of a process after it is started.
- Suspended: The process is running, but suspended due to an `EVENTACTIONS SUSPEND` action. In a suspended state, the process is not active, and no data can be processed, but the state of the current run is preserved and can be continued by issuing the `RESUME` command. The RBA in the `INFO` command reflects the last checkpointed position before the suspend action. To determine whether the state is active or suspended, issue the `SEND EXTRACT` command with the `STATUS` option.

Syntax

```
STATUS EXTRACT group_name [, TASKS | ALLPROCESSES]
```

group_name
The name of an Extract group or a wildcard (*) to specify multiple groups. For example, `T*` returns status for all Extract groups whose names begin with T.

`TASKS`
Displays status only for Extract tasks. By default, tasks are not displayed unless you specify a single Extract group (without wildcards).

`ALLPROCESSES`
Displays status for all Extract groups, including tasks.

Examples

```
STATUS EXTRACT exte
```

```
STATUS EXTRACT ext*
```

STATUS REPLICAT

Use `STATUS REPLICAT` to determine whether or not Replicat is running. There are the following four possible statuses:

Abended
The process abnormally ended.

Running
Means one of the following:

- **Active:** Running and processing (or able to process) data. This is the normal state of a process after it is started.

- **Suspended:** The process is running though suspended due to an `EVENTACTIONS SUSPEND` action. In a suspended state, the process is not active, and no data can be processed, but the state of the current run is preserved and can be continued by issuing the `RESUME` command in GGSCI. The RBA in the `INFO` command reflects the last checkpointed position before the suspend action. To determine whether the state is active or suspended, issue a `SEND EXTRACT|REPLICAT` *`group_name`* `STATUS` command. For more information, see SEND EXTRACT or SEND REPLICAT.

Starting
The process is starting.

Stopped
The process was stopped.

Syntax

```
STATUS REPLICAT group_name
[, TASKS]
[, ALLPROCESSES]
```

`group_name`
The name of a Replicat group or a wildcard (*) to specify multiple groups. For example, `T*` shows status for all Replicat groups whose names begin with T.

`TASKS`
Displays status only for Replicat tasks. By default, tasks are not displayed unless you specify a single Replicat group (without wildcards).

`ALLPROCESSES`
Displays status for all Replicat groups, including tasks.

Examples

```
STATUS REPLICAT repe
```

```
STATUS REPLICAT rep*
```

STATUS SERVICE

Use `STATUS SERVICE` to display status of the specified Oracle GoldenGate services.

Syntax

```
STATUS SERVICE service-name-wildcard
```

`service-name-wildcard`
The name of an service or a wildcard (*) to specify multiple services. For example, `T*` statuses all services whose names begin with T.

STOP DEPLOYMENT

Use `STOP DEPLOYMENT` to stop one or more deployments.

Syntax

```
STOP DEPLOYMENT deployment_name_wildcard [!]
```

deployment_name_wildcard
The name of the deployment or a wildcard (*) to specify multiple deployments. For example, `P*` sends the command to all deployments whose group names begin with P.

!
(Exclamation point) Stops the deployment immediately. The transaction is ended and the process terminates.

Example

```
STOP DEPLOYMENT production
```

STOP ER

Use the `STOP ER` to stop multiple Extract and Replicat groups as a unit. Use it with wildcards to affect every Extract and Replicat group that satisfies the wildcard. For descriptions and optional parameters for this command, see `STOP EXTRACT`.

Syntax

```
STOP  ER group_name [!]
```

group_name
The wildcard specification for the groups that you want to affect with the command. Oracle GoldenGate automatically increases internal storage to track up to 100,000 wildcard entries.

!
(Exclamation point) Stops the Extracts and Replicats immediately. The transaction is ended and the process terminates.

Examples

```
STOP ER * !
```

STOP EXTRACT

Use `STOP EXTRACT` to stop Extract gracefully. The command preserves the state of synchronization for the next time Extract starts, and it ensures that Manager does not automatically start Extract.

If there are open, long-running transactions when you issue `STOP EXTRACT`, you might be advised of the oldest transaction log file that will be needed for that transaction when Extract is restarted. You can use the `SEND EXTRACT` option of `SHOWTRANS` to view details and data of those

transactions and then, if desired, use the SKIPTRANS or FORCETRANS options to skip the transaction or force it to be written as a committed transaction to the trail, see SEND EXTRACT.

Syntax

```
STOP EXTRACT group_name [!]
```

group_name
The name of an Extract group or a wildcard (*) to specify multiple groups. For example, T* stops all Extract processes for groups whose names begin with T.

!
(Exclamation point) Stops Extract immediately. The transaction is ended and the process terminates.

Examples

```
STOP EXTRACT exte


STOP EXTRACT * !
```

STOP DISTPATH

Use STOP DISTPATH to stop a distribution path and attempt to cleanup the resource. To confirm that the distribution path has stopped, use the INFO DISTPATH command. To change the distribution path start point, use the ALTER DISTPATH command.

Syntax

```
STOP DISTPATH path_name [ALL | !]
```

path_name
The name of the distribution path.

ALL
Use to stop all distribution paths.

!
(Exclamation point) Stops a distribution path immediately. The transaction is terminated and the path is stopped forcefully.

STOP RECVPATH

Use STOP RECVPATH to stop a target-initiated distribution path in the Receiver Service. To confirm that the distribution path has stopped, use the INFO RECVPATH command. To change the distribution path start point, use the ALTER RECVPATH command.

Syntax

```
STOP RECVPATH path_name [ALL | !]
```

path_name
The name of the deployment to specify multiple deployments. For example, `P*` sends the command to all deployments whose group names begin with P.

`ALL`
Use to stop all target-initiated distribution paths.

`!`
(Exclamation point) Stops a target-initiated distribution path immediately. The transaction is ended and the process terminates.

STOP REPLICAT

Use `STOP REPLICAT` to stop Replicat cleanly. This command preserves the state of synchronization for the next time Replicat starts, and it ensures that Manager does not automatically start Replicat.

In a clean shutdown of a coordinated Replicat, the coordinator thread attempts to stop all of the threads on the same transaction boundary. If the shutdown of a coordinated Replicat is not clean, the threads may stop at different positions in the trail file. If this happens, `START REPLICAT` writes a warning if the parameter file was changed since the prior run and raises an error if the number of threads was changed. To resolve these problems and start Replicat again, see About Coordinated Replicat.

Syntax

```
STOP REPLICAT group_name [!]
```

group_name
The name of a Replicat group or a wildcard (*) to specify multiple groups. For example, `T*` stops all Replicat groups whose names begin with T.

`!`
(Exclamation point) Stops Replicat immediately. The transaction is ended and the process terminates.

Example

```
STOP REPLICAT repe


STOP REPLICAT * !
```

STOP SERVICE

Use `STOP SERVICE` to stop the specified Oracle GoldenGate services.

Syntax

```
STOP SERVICE service_name_wildcard service_name_wildcard
```

The name of a service or a wildcard (*) to specify multiple services. For example, `T*` stops all services whose names begin with T.

SYNCHRONIZE REPLICAT

Valid for coordinated, integrated, and parallel Replicat. Use `SYNCHRONIZE REPLICAT` to return all of the threads of a Replicat to the same position in the trail file after an unclean shutdown. This position is the maximum checkpoint position of all of the threads, in other words, the most recent trail record processed among all of the threads. When `SYNCHRONIZE REPLICAT` is issued, all threads are started and allowed to process transactions until they reach the maximum checkpoint position, and then Replicat stops.

For more information about how to use `SYNCHRONIZE REPLICAT` to recover a coordinated Replicat after an unclean shutdown, or to enable repartitioning of data among different threads, see About Coordinated Replicat.

Syntax

```
SYNCHRONIZE REPLICAT group_name
```

group_name
The name of a Replicat group or a wildcard (*) to specify multiple groups. For example, `T*` synchronizes the threads of all Replicat groups whose names begin with T. The threads synchronize to the same position within their group, not to the same position across all Replicat groups being synchronized with this command.

Example

```
SYNCHRONIZE REPLICAT repe
```

UNDELETE MASTERKEY

Use the `UNDELETE MASTERKEY` command to remove the deletion mark from a master key version, thus retaining that version if the `PURGE WALLET` command is used. Only one version can be unmarked per `UNDELETE MASTERKEY` command. See DELETE MASTERKEY to mark a version of a master key for deletion.

The `OPEN WALLET` command must be used before using this command or any of the commands that add, renew, or delete the master keys in the wallet.

Syntax

```
UNDELETE MASTERKEY VERSION version
```

VERSION *version*
The version that is to be unmarked for deletion.

Example

This command unmarks version 3 of the master key and returns a message similar to the one shown.

```
UNDELETE MASTERKEY VERSION 3
Version 3 of Masterkey 'OGG_DEFAULT_MASTERKEY' undeleted from wallet
at location './ wallet'.
```

Or

```
UNDELETE MASTERKEY VERSION 3
Version 3 of Masterkey 'OGG_DEFAULT_MASTERKEY' undeleted from wallet at
location './dirwlt'.
```

UNREGISTER EXTRACT

Valid for Oracle and PostgreSQL.

Use `UNREGISTER EXTRACT` to remove the registration of an Extract group from an Oracle AI Database. `UNREGISTER EXTRACT` is valid only for a primary Extract group.

To register an Extract group with the database, use the `REGISTER EXTRACT` command.

To upgrade an Extract from classic capture mode to integrated capture mode, use the `ALTER EXTRACT` command.

For PostgreSQL, a replication slot is dropped in the connected database for PostgreSQL. This command ensures that the PostgreSQL database overwrites the existing transaction log or may archive the log . After deleting the Extract, the command must be run.

Syntax

```
UNREGISTER EXTRACT group_name
```

group_name
The name of the Extract group that is to be unregistered from the database. Do not use a wildcard. This group must currently be registered with the database.

DATABASE
(Oracle only) Disables integrated capture mode for the Extract group.
(Oracle only) This command removes the database capture (mining) server that has the same name as the Extract group. For additional information about support for, and configuration of, the Extract capture modes.
Before using `UNREGISTER EXTRACT` with `DATABASE`, do the following:

1. Stop Extract with the `STOP EXTRACT` command.
2. Log in to the mining database with the `DBLOGIN` or `MININGDBLOGIN` command with the privileges granted in the `dbms_goldengate_auth.grant_admin_privilege` procedure. For local capture, `DBLOGIN` is required. For downstream capture, `DBLOGIN` and `MININGDBLOGIN` are both required.
3. Delete the Extract group with `DELETE EXTRACT`.

Examples

```
UNREGISTER EXTRACT exte
```

```
UNREGISTER EXTRACT exte DATABASE
```

```
UNREGISTER EXTRACT exte with DATABASE PostgresDB
```

UPGRADE CHECKPOINTTABLE

Not valid for Replicat for Java, Oracle GoldenGate Applications Adapter, or Oracle GoldenGate Big Data.

Use the `UPGRADE CHECKPOINTTABLE` command to add a supplemental checkpoint table when upgrading Oracle GoldenGate.

Syntax

```
UPGRADE CHECKPOINTTABLE [[container. | catalog.]owner.table]
```

container. | catalog.
The Oracle pluggable database. If this option is omitted, the catalog or pluggable database defaults to the one that is associated with the `SOURCEDB`, `USERID`, or `USERIDALIAS` portion of the `DBLOGIN` command (depending on the database).

owner.table
The owner and name of the checkpoint table. An owner and name are not required if they are the same as those specified with the `CHECKPOINTTABLE` parameter in the `GLOBALS` file.

Example

```
UPGRADE CHECKPOINTTABLE ggadmin.ggs_checkpoint
```

UPGRADE HEARTBEATTABLE

Valid for all supported databases. See the Certification Matrix for details on supported databases.

Use `UPGRADE HEARTBEATTABLE` when upgrading Oracle GoldenGate from a prior release, to enable any new heartbeat functionality available in the current release.

This command requires a `DBLOGIN`. On a CDB database for Oracle, a PDB login is required.

Oracle GoldenGate for Oracle AI Database simplifies the administration of the heartbeat table by eliminating the need for `GGSCHEMA` or `HEARTBEATTABLE` parameter. In case of CDB root Extract, `GGSCHEMA` is used.

Syntax

```
UPGRADE HEARTBEATTABLE
```

VALIDATE AUTHORIZATIONPROFILE

Use this command to validate the authorization profile. Validating an authorization profile carries out a connection and configuration test. This test checks that the services can establish a secure communication with the described IDP tenant and that the IDP application, with the exception of redirect URIs, is configured correctly.

If validation fails, information on what failed in the test gets printed.

Syntax

```
VALIDATE AUTHORIZATIONPROFILE profile-name
         DEPLOYMENT deployment-name
```

profile-name
Name of profile to be validated.

deployment-name
Name of the deployment where the profile resides.

Example

The following example validates the profile (`testProfile`) in Service Manager.

```
Validate testProfile in ServiceManager
VALIDATE AUTHORIZATIONPROFILE testProfile DEPLOYMENT ServiceManager
```

VERSIONS

Use `VERSIONS` to display operating system and database version information. For ODBC connections, the driver version is also displayed. To include database information in the output, issue a `DBLOGIN` command before issuing `VERSIONS` to establish a database connection.

Syntax

```
VERSIONS
```

VIEW DISCARD

Use `VIEW DISCARD` to display the discard file that is generated by Extract or Replicat. The `SET PAGER` value is used to determine pagination of the output.

Syntax

```
VIEW DISCARD report-name
```

report-name

The name of the report to display the discard file. For `EXTRACT "EXX"`, these report names are valid:

- `EXX`
- `EXX0`
- `EXX1`
- `...`
- `EXX9`

No other values are valid.

VIEW ENCKEYS

Use `VIEW ENCKEYS` to display the contents of the `ENCKEYS` file in read-only mode on-screen.

Syntax

```
VIEW ENCKEYS
```

VIEW GLOBALS

Use `VIEW GLOBALS` to display the contents of the `GLOBALS` parameter file in read-only mode on-screen. The `SET PAGER` value is used to determine pagination of the output.

Syntax

```
VIEW GLOBALS
```

VIEW MESSAGES

Use `VIEW MESSAGES` to display the message log file, `ggserr.log`. The `SET PAGER` value is used to determine pagination of the output.

Syntax

```
VIEW MESSAGES
```

VIEW PARAMS

Use `VIEW PARAMS` to view the contents of a parameter file.

> ⚠ **Caution**
>
> Do not use this command to view a parameter file that is in a character set other than that of the local operating system (such as one where the `CHARSET` option was used to specify a different character set). The contents may become corrupted. View the parameter file from outside the command line interface

Syntax

```
VIEW PARAMS {group_name | file_name}
```

group_name
Shows the parameter file for the specified Extract or Replicat group.

file_name
Shows the specified file. The default sub-directory is used if no path is specified. If the parameter file resides in a directory other than the default, specify the full path name.

Examples

```
VIEW PARAMS exte
```

```
VIEW PARAMS c:\lpparms\repe.prm
```

VIEW REPORT

Use `VIEW REPORT` to view the process report or the discard file that is generated by Extract or Replicat. Each process generates a new report and discard file upon startup.

Reports and discard files are aged whenever a process starts. Old files are appended with a sequence number, for example `exte.rpt`, `exte1.rpt`, and so forth, or `discard0.dsc`, `discard1.dsc`, and so forth.

Syntax

```
VIEW REPORT group_name[version]
```

group_name
The name of the Extract or Replicat group. The command assumes the report file named *group*`.rpt` or the discard file named *group*`.dsc` in the Oracle GoldenGate default subdirectory. Use the relative file name if stored in the default location, or the full path name if not stored in the default location

version
To view the current report or discard file, use the command without this option. Specify the report you want to see by number.

Examples

View the most recent (active) report for the `exte` group.

```
VIEW REPORT exte
```

View the second most recent report.

```
VIEW REPORT exte2
```

View the eleventh most recent report.

```
VIEW REPORT exte11
```