# 8 ORDS\_SECURITY PL/SQL Package Reference

## 8.1 JWT Profile actions

### 8.1.1 Create an OAuth2.0 JWT Profile

#### 8.1.1.1 Format

PROCEDURE CREATE\_JWT\_PROFILE (  
 P\_ISSUER IN OAUTH\_JWT\_PROFILE.ISSUER%TYPE,  
 P\_AUDIENCE IN OAUTH\_JWT\_PROFILE.AUDIENCE%TYPE,  
 P\_JWK\_URL IN OAUTH\_JWT\_PROFILE.JWK\_URL%TYPE,  
 P\_DESCRIPTION IN OAUTH\_JWT\_PROFILE.DESCRIPTION%TYPE DEFAULT NULL,  
 P\_ALLOWED\_SKEW IN OAUTH\_JWT\_PROFILE.ALLOWED\_SKEW%TYPE DEFAULT NULL,  
 P\_ALLOWED\_AGE IN OAUTH\_JWT\_PROFILE.ALLOWED\_AGE%TYPE DEFAULT NULL  
);

#### 8.1.1.2 Description

This procedure creates an OAuth2 JWT Profile for *your* schema.[[1]](#footnote-21) The profile is used to validate JWTs, as well as validate that the subject property in the JWT has the appropriate scope (as defined in the related ORDS privilege) for this schema.

#### 8.1.1.3 Parameters

| Parameter | Description | Notes |
| --- | --- | --- |
| p\_issuer | The issuer of acceptable JWT access tokens. This value must match the “iss” claim provided in the JWT. |  |
| p\_audience | The audience of acceptable JWT access tokens. This value must match the “aud” claim provided in the JWT. |  |
| p\_jwk\_url | This is the url to the jwk(s) used to validate acceptable JWT access tokens. It must start with “https://” |  |
| p\_description | A description of the JWT Profile. | - *Optional* |
| p\_allowed\_skew | The number of seconds allowed to skew time claims provided in the JWT. | - *Optional* |
| p\_allowed\_age | The maximum allowed age of a JWT in seconds, regardless of expired claim. | - *Optional* |

:bulb: **Tip:** You can review existing a JWT Profile with the USER\_ORDS\_JWT\_PROFILE View. Use the following SQL query SELECT \* USER\_ORDS\_JWT\_PROFILE;, or fully-qualified version SELECT \* FROM ORDS\_METADATA.USER\_ORDS\_JWT\_PROFILE;.

#### 8.1.1.4 Usage Notes

If a JWT profile already exists, then it must be deleted first. For this operation to take effect, include the COMMIT statement after calling the ORDS\_SECURITY.DELETE\_JWT\_PROFILE; procedure.

You can execute this procedure seperately or from with the same block as the ORDS\_SECURITY.CREATE\_JWT\_PROFILE; procedure.

*Executing the ORDS\_SECURITY.DELETE\_JWT\_PROFILE; separately*

BEGIN  
 ORDS\_SECURITY.DELETE\_JWT\_PROFILE();  
 COMMIT;  
END;

*As a simple EXECUTE command:*

EXECUTE ORDS\_SECURITY.DELETE\_JWT\_PROFILE;  
COMMIT;

*Included in the same block as the CREATE\_JWT\_PROFLE procedure.*

BEGIN  
 ORDS\_SECURITY.DELETE\_JWT\_PROFILE;  
 ORDS\_SECURITY.CREATE\_JWT\_PROFILE(  
 p\_issuer => 'https://identity.oraclecloud.com/',  
 p\_audience => 'ords/myapplication/api' ,  
 p\_jwk\_url =>'https://  
 idcs-10a10a10a10a10a10a10a10a.identity.oraclecloud.com/admin/v1/SigningCert/  
 jwk'  
 );  
 COMMIT;  
END;  
/

#### 8.1.1.5 Examples

##### Oracle Identity Access Management

DECLARE  
 L\_P\_ISSUER VARCHAR2(200) := 'https://identity.oraclecloud.com/';  
 L\_P\_AUDIENCE VARCHAR2(200) := 'ords/';  
 L\_P\_JWK\_URL VARCHAR2(200) := 'https://idcs-66de820ed85f41f0805119c5967689b2.identity.oraclecloud.com:443/admin/v1/SigningCert/jwk'  
 ;  
 L\_P\_DESCRIPTION VARCHAR2(200) := 'Example JWT Client Profile';  
 L\_P\_ALLOWED\_SKEW NUMBER;  
 L\_P\_ALLOWED\_AGE NUMBER;  
BEGIN  
 ORDS\_METADATA.ORDS\_SECURITY.CREATE\_JWT\_PROFILE(  
 P\_ISSUER => L\_P\_ISSUER,  
 P\_AUDIENCE => L\_P\_AUDIENCE,  
 P\_JWK\_URL => L\_P\_JWK\_URL,  
 P\_DESCRIPTION => L\_P\_DESCRIPTION,  
 P\_ALLOWED\_SKEW => L\_P\_ALLOWED\_SKEW,  
 P\_ALLOWED\_AGE => L\_P\_ALLOWED\_AGE  
 );  
END;

##### Microsoft Entra

BEGIN  
 ORDS\_METADATA.ORDS\_SECURITY.CREATE\_JWT\_PROFILE(  
 P\_ISSUER => https://login.microsoftonline.com/[Your Microsoft Tenancy ID]/v2.0,  
 P\_AUDIENCE => '1234abcd-ef56-gh78-ij91-123456abcdefg',  
 P\_JWK\_URL => 'https://login.microsoftonline.common/discovery/v2.0/keys',  
 P\_DESCRIPTION => Example JWT Client Profile  
 );   
END;

:memo: **Note:** Identify providers may differ on how certain parameters should be structured. For example, OCI Identity Access Management uses the Primary Audience of an Integrated Application and requires a trailing slash for the AUD claim, whereas Microsoft Entra uses the Application (client) ID as the AUD claim. Please refer to your Identity Provider’s documentation for guidance on how claims should be structured.

### 8.1.2 Delete an OAuth2.0 JWT Profile

#### 8.1.2.1 Format

PROCEDURE DELETE\_JWT\_PROFILE;

#### 8.1.2.2 Description

This procedure deletes an existing JWT Profile. Once a JWT profile has been deleted, JWT bearer tokens will no longer be accepted when authorizing requests to those protected resources.

#### 8.1.2.3 Usage notes

For this operation to take effect, use the COMMIT statement after calling this method.

#### 8.1.2.4 Examples

*As an PL/SQL Anonymous Block:*

BEGIN  
 ORDS\_SECURITY.DELETE\_JWT\_PROFILE;  
COMMIT;  
END

*As a simple EXECUTE command:*

EXECUTE ORDS\_SECURITY.DELETE\_JWT\_PROFILE;  
COMMIT;

## 8.2 OAuth2.0 Client actions

### 8.2.1 Delete an OAuth2.0 Client registration

An OAuth2.0 client configuration can be delete using one of three “keys.” Deleting by a client’s name is the easist method.

#### 8.2.1.1 Delete an OAuth2.0 Client registration *by name*

##### 8.2.1.1.1 Format

PROCEDURE DELETE\_CLIENT (  
 P\_NAME IN VARCHAR2  
);

PROCEDURE DELETE\_CLIENT (  
 P\_CLIENT\_KEY IN ORDS\_TYPES.T\_CLIENT\_KEY  
);

##### 8.2.1.1.2 Description

Deletes an OAuth2.0 client registration using the ID of the OAuth2.0 client name.[[2]](#footnote-36)

##### 8.2.1.1.3 Parameters

| Parameter | Description | Notes |
| --- | --- | --- |
| p\_name | The name of the client registration to be deleted. This value must not be null. |  |

##### 8.2.1.1.4 Usage notes

Use the example COMMIT statement after calling this method for this operation to take effect.

##### Examples

BEGIN  
 ORDS\_SECURITY.DELETE\_CLIENT(P\_NAME => 'test\_client');  
COMMIT;  
  
END;  
/

BEGIN  
ORDS\_SECURITY.DELETE\_CLIENT(  
 p\_client\_key => ords\_types.t\_client\_key(p\_name=>'test\_client')  
);  
COMMIT;  
END;  
/

#### 8.2.1.2 Delete an OAuth2.0 Client registration *by id*

##### 8.2.1.2.1 Format

PROCEDURE delete\_client(  
 p\_client\_key IN ords\_types.t\_client\_key  
 );

##### 8.2.1.2.2 Description

Deletes an OAuth2.0 client registration using the ID of the client.[[3]](#footnote-43)

:memo: **Note:** The ID and the Client ID are different values. The ID of the client is the unique identifer used for functions and identification *within* the database. Whereas the Client ID is the ID that is associated with a registered client application (i.e. the value ususally stored in your application’s environment variables file).

##### 8.2.1.2.3 Parameters

| Parameter | Description | Notes |
| --- | --- | --- |
| p\_client\_key | The key of the client registration to be deleted. A minimum of one key must be supplied. | *One of:* |

##### 8.2.1.2.4 Usage notes

Use the example COMMIT statement after calling this method for this operation to take effect.

##### 8.2.1.2.5 Examples

BEGIN  
PROCEDURE DELETE\_CLIENT (  
 P\_CLIENT\_KEY IN ORDS\_TYPES.T\_CLIENT\_KEY(p\_id => 10604)  
);  
COMMIT;  
END;  
/

#### 8.2.1.3 Delete an OAuth2.0 Client registration *by client\_id*

##### 8.2.1.3.1 Format

PROCEDURE ORDS\_SECURITY.DELETE\_CLIENT(  
 p\_client\_key IN ords\_types.t\_client\_key  
);

##### 8.2.1.3.2 Description[[4]](#footnote-50)

:memo: **Note:** The ID and the Client ID are different values. The ID of the client is the unique identifer used for functions and identification *within* the database. Whereas the Client ID is the ID that is associated with a registered client application (i.e. the value ususally stored in your application’s environment variables file).

##### 8.2.1.3.3 Parameters

| Parameter | Description | Notes |
| --- | --- | --- |
| p\_client\_key | The key of the client registration to be deleted. A minimum of one key must be supplied. | *One of:* |

##### 8.2.1.3.4 Usage notes

Use the example COMMIT statement after calling this method for this operation to take effect.

##### 8.2.1.3.5 Examples

BEGIN  
PROCEDURE DELETE\_CLIENT (  
 P\_CLIENT\_KEY IN ORDS\_TYPES.T\_CLIENT\_KEY(  
 p\_client\_id => '7a8ZBfjTmkxX6fSBBLIPig..')  
);  
COMMIT;  
END;  
/

### 8.2.2 Importing an OAuth2.0 client

/\*\* \* Import an OAuth client  No secret will be registered by the method. Use register\_client\_secret or \* rotate\_client\_secret subsequently to register a secret.  @param p\_name Human readable name for the client, displayed to the end user \* during the approval phase of three-legged OAuth. \* This value must not be null \* @param p\_grant\_type Must be one of: ‘authorization\_code’, ‘implicit’ or \* ‘client\_credentials’. \* @param p\_client\_id The original generated client identifier @see ORDS\_EXPORT \* When null, a new client identifier is generated \* @param p\_privilege\_names Names of the privileges that the client wishes to \* access. Each privilege name must be separated by a comma character. \* @param p\_description Human readable description of the purpose of the \* client, displayed to the end user during the \* approval phase of three-legged OAuth. May be null \* if p\_grant\_type == ‘client\_credentials’, non null \* otherwise. \* @param p\_redirect\_uri Client controlled URI to which redirect containing \* OAuth access token/error will be sent. May be null \* if p\_grant\_type == ‘client\_credentials’, non null \* otherwise. \* @param p\_support\_email Support e-mail for client’s users \* @param p\_support\_uri Support URI for client’s users \* @param p\_token\_duration Duration of the access token (bearer) in seconds. Null uses ORDS instance default value \* Which can be set by property or 3600 seconds by default. \* @param p\_refresh\_duration Duration of refresh token in seconds. \* Null uses ORDS instance default value set by a property or 86400 seconds. \* @param p\_code\_duration Duration of the code token in seconds (only applicable when grant\_type is authorization \* code. If the value is set to null or the grant\_type is not authorization\_code. The lifetime \* will be the one defined in the ords instance access\_token/12 (300 by default). \* @return The client key (including client\_id). \*/ FUNCTION import\_client( p\_name IN VARCHAR2, p\_grant\_type IN VARCHAR2, p\_description IN VARCHAR2 DEFAULT NULL, p\_client\_id IN VARCHAR2 DEFAULT NULL, p\_privilege\_names IN VARCHAR2 DEFAULT NULL, p\_origins\_allowed IN VARCHAR2 DEFAULT NULL, p\_redirect\_uri IN VARCHAR2 DEFAULT NULL, p\_support\_email IN VARCHAR2 DEFAULT NULL, p\_support\_uri IN VARCHAR2 DEFAULT NULL, p\_token\_duration IN NUMBER DEFAULT NULL, p\_refresh\_duration IN NUMBER DEFAULT NULL, p\_code\_duration IN NUMBER DEFAULT NULL ) RETURN ords\_types.t\_client\_key;

/\*\* \* Import an OAuth client  No secret will be registered by the method. Use register\_client\_secret or \* rotate\_client\_secret subsequently to register a secret.  @param p\_name Human readable name for the client, displayed to the end user \* during the approval phase of three-legged OAuth. \* This value must not be null \* @param p\_grant\_type Must be one of: ‘authorization\_code’, ‘implicit’ or \* ‘client\_credentials’. \* @param p\_client\_id The original generated client identifier @see ORDS\_EXPORT \* When null, a new client identifier is generated \* @param p\_privilege\_names Names of the privileges that the client wishes to \* access. Each privilege name must be separated by a comma character. \* @param p\_description Human readable description of the purpose of the \* client, displayed to the end user during the \* approval phase of three-legged OAuth. May be null \* if p\_grant\_type == ‘client\_credentials’, non null \* otherwise. \* @param p\_redirect\_uri Client controlled URI to which redirect containing \* OAuth access token/error will be sent. May be null \* if p\_grant\_type == ‘client\_credentials’, non null \* otherwise. \* @param p\_support\_email Support e-mail for client’s users \* @param p\_support\_uri Support URI for client’s users \* @param p\_token\_duration Duration of the access token (bearer) in seconds. Null uses ORDS instance default value \* Which can be set by property or 3600 seconds by default. \* @param p\_refresh\_duration Duration of refresh token in seconds. \* Null uses ORDS instance default value set by a property or 86400 seconds. \* @param p\_code\_duration Duration of the code token in seconds (only applicable when grant\_type is authorization \* code. If the value is set to null or the grant\_type is not authorization\_code. The lifetime \* will be the one defined in the ords instance access\_token/12 (300 by default). \*/ PROCEDURE import\_client( p\_name IN VARCHAR2, p\_grant\_type IN VARCHAR2, p\_description IN VARCHAR2 DEFAULT NULL, p\_client\_id IN VARCHAR2 DEFAULT NULL, p\_privilege\_names IN VARCHAR2 DEFAULT NULL, p\_origins\_allowed IN VARCHAR2 DEFAULT NULL, p\_redirect\_uri IN VARCHAR2 DEFAULT NULL, p\_support\_email IN VARCHAR2 DEFAULT NULL, p\_support\_uri IN VARCHAR2 DEFAULT NULL, p\_token\_duration IN NUMBER DEFAULT NULL, p\_refresh\_duration IN NUMBER DEFAULT NULL, p\_code\_duration IN NUMBER DEFAULT NULL );

### 8.2.3 Register an OAuth2.0 client

/\*\* \* Register an OAuth client  @param p\_name Human readable name for the client, displayed to the end user \* during the approval phase of three-legged OAuth. \* This value must not be null \* @param p\_grant\_type Must be one of: ‘authorization\_code’, ‘implicit’ or \* ‘client\_credentials’. \* @param p\_description Human readable description of the purpose of the \* client, displayed to the end user during the \* approval phase of three-legged OAuth. May be null \* if p\_grant\_type == ‘client\_credentials’, non null \* otherwise. \* @param p\_client\_secret The client secret. Any fields can be set except issued\_on. \* (DEFAULT: no secret is registered). \* @param p\_privilege\_names Names of the privileges that the client wishes to \* access. Each privilege name must be separated by a comma character. \* @param p\_origins\_allowed allowed origins \* @param p\_redirect\_uri Client controlled URI to which redirect containing \* OAuth access token/error will be sent. May be null \* if p\_grant\_type == ‘client\_credentials’, non null \* otherwise. \* @param p\_support\_email Support e-mail for client’s users \* @param p\_support\_uri Support URI for client’s users \* @param p\_token\_duration Bearer duration in seconds. \* @param p\_refresh\_duration Refresh duration in seconds where applicable. \* @param p\_code\_duration Code duration in seconds. \* @return The client key (id|name|client\_id) and client\_secret, if any, of the registered client \*/ FUNCTION register\_client( p\_name IN VARCHAR2, p\_grant\_type IN VARCHAR2, p\_description IN VARCHAR2 DEFAULT NULL, p\_client\_secret IN ords\_types.t\_client\_secret DEFAULT ords\_constants.oauth\_client\_secret\_skip, p\_privilege\_names IN VARCHAR2 DEFAULT NULL, p\_origins\_allowed IN VARCHAR2 DEFAULT NULL, p\_redirect\_uri IN VARCHAR2 DEFAULT NULL, p\_support\_email IN VARCHAR2 DEFAULT NULL, p\_support\_uri IN VARCHAR2 DEFAULT NULL, p\_token\_duration IN NUMBER DEFAULT NULL, p\_refresh\_duration IN NUMBER DEFAULT NULL, p\_code\_duration IN NUMBER DEFAULT NULL ) RETURN ords\_types.t\_client\_credentials;

/\*\* \* Register an OAuth client  No secret will be registered for this client. Use register\_client\_secret or \* rotate\_client\_secret to register a secret.  @param p\_name Human readable name for the client, displayed to the end user \* during the approval phase of three-legged OAuth. \* This value must not be null \* @param p\_grant\_type Must be one of: ‘authorization\_code’, ‘implicit’ or \* ‘client\_credentials’. \* @param p\_description Human readable description of the purpose of the \* client, displayed to the end user during the \* approval phase of three-legged OAuth. May be null \* if p\_grant\_type == ‘client\_credentials’, non null \* otherwise. \* @param p\_privilege\_names Names of the privileges that the client wishes to \* access. Each privilege name must be separated by a comma character. \* @param p\_origins\_allowed allowed origins \* @param p\_redirect\_uri Client controlled URI to which redirect containing \* OAuth access token/error will be sent. May be null \* if p\_grant\_type == ‘client\_credentials’, non null \* otherwise. \* @param p\_support\_email Support e-mail for client’s users \* @param p\_support\_uri Support URI for client’s users \* @param p\_token\_duration Bearer duration in seconds. \* @param p\_refresh\_duration Refresh duration in seconds where applicable. \* @param p\_code\_duration Code duration in seconds. \*/ PROCEDURE register\_client( p\_name IN VARCHAR2, p\_grant\_type IN VARCHAR2, p\_description IN VARCHAR2 DEFAULT NULL, p\_privilege\_names IN VARCHAR2 DEFAULT NULL, p\_origins\_allowed IN VARCHAR2 DEFAULT NULL, p\_redirect\_uri IN VARCHAR2 DEFAULT NULL, p\_support\_email IN VARCHAR2 DEFAULT NULL, p\_support\_uri IN VARCHAR2 DEFAULT NULL, p\_token\_duration IN NUMBER DEFAULT NULL, p\_refresh\_duration IN NUMBER DEFAULT NULL, p\_code\_duration IN NUMBER DEFAULT NULL );

### 8.2.4 Updating an OAuth2.0 client registration

/\*\* \* Update an OAuth client registration.  @param p\_client\_key The key (id|name|client\_id) of the client to be modified. \* A minimum of one key must be supplied. \* @param p\_new\_name Human readable name for the client, displayed to the end user \* during the approval phase of three-legged OAuth. \* When null, the old name is preserved. \* @param p\_description Human readable description of the purpose of the \* client, displayed to the end user during the \* approval phase of three-legged OAuth. May be null \* if p\_grant\_type == ‘client\_credentials’, non null \* otherwise. \* @param p\_origins\_allowed allowed origins \* @param p\_redirect\_uri Client controlled URI to which redirect containing \* OAuth access token/error will be sent. May be null \* if p\_grant\_type == ‘client\_credentials’, non null \* otherwise. \* @param p\_support\_email Support e-mail for client’s users \* @param p\_support\_uri Support URI for client’s users \* @return The client key (id|name|client\_id) of the updated client. \*/ FUNCTION update\_client( p\_client\_key IN ords\_types.t\_client\_key, p\_new\_name IN VARCHAR2 DEFAULT NULL, p\_description IN VARCHAR2, p\_origins\_allowed IN VARCHAR2, p\_redirect\_uri IN VARCHAR2, p\_support\_email IN VARCHAR2, p\_support\_uri IN VARCHAR2 ) RETURN ords\_types.t\_client\_key;

/\*\* \* Update an OAuth client registration.  @param p\_name The name of the client to be modified. \* This value must not be null. \* @param p\_new\_name Human readable name for the client, displayed to the end user \* during the approval phase of three-legged OAuth. \* When null, the old name is preserved. \* @param p\_description Human readable description of the purpose of the \* client, displayed to the end user during the \* approval phase of three-legged OAuth. May be null \* if p\_grant\_type == ‘client\_credentials’, non null \* otherwise. \* @param p\_origins\_allowed allowed origins \* @param p\_redirect\_uri Client controlled URI to which redirect containing \* OAuth access token/error will be sent. May be null \* if p\_grant\_type == ‘client\_credentials’, non null \* otherwise. \* @param p\_support\_email Support e-mail for client’s users \* @param p\_support\_uri Support URI for client’s users \*/ PROCEDURE update\_client( p\_name IN VARCHAR2, p\_new\_name IN VARCHAR2 DEFAULT NULL, p\_description IN VARCHAR2, p\_origins\_allowed IN VARCHAR2, p\_redirect\_uri IN VARCHAR2, p\_support\_email IN VARCHAR2, p\_support\_uri IN VARCHAR2 );

/\*\* \* Update an OAuth client registration.  @param p\_client\_key The key (id|name|client\_id) of the client to be modified. \* A minimum of one key must be supplied. \* @param p\_new\_name Human readable name for the client, displayed to the end user \* during the approval phase of three-legged OAuth. \* When null, the old name is preserved. \* @param p\_description Human readable description of the purpose of the \* client, displayed to the end user during the \* approval phase of three-legged OAuth. May be null \* if p\_grant\_type == ‘client\_credentials’, non null \* otherwise. \* @param p\_privilege\_names Names of the privileges that the client wishes to \* access. Each privilege name must be separated by a comma character. \* @param p\_origins\_allowed allowed origins \* @param p\_redirect\_uri Client controlled URI to which redirect containing \* OAuth access token/error will be sent. May be null \* if p\_grant\_type == ‘client\_credentials’, non null \* otherwise. \* @param p\_support\_email Support e-mail for client’s users \* @param p\_support\_uri Support URI for client’s users \* @param p\_token\_duration Duration of the access token (bearer) in seconds. Null uses ORDS instance default value \* Which can be set by property or 3600 seconds by default. \* @param p\_refresh\_duration Duration of refresh token in seconds. \* Null uses ORDS instance default value set by a property or 86400 seconds. \* @param p\_code\_duration Duration of the code token in seconds (only applicable when grant\_type is authorization \* code. If the value is set to null or the grant\_type is not authorization\_code. The lifetime \* will be the one defined in the ords instance access\_token/12 (300 by default). \* @return The client key (id|name|client\_id) of the updated client. \*/ FUNCTION update\_client( p\_client\_key IN ords\_types.t\_client\_key, p\_new\_name IN VARCHAR2 DEFAULT NULL, p\_description IN VARCHAR2, p\_privilege\_names IN VARCHAR2, p\_origins\_allowed IN VARCHAR2, p\_redirect\_uri IN VARCHAR2, p\_support\_email IN VARCHAR2, p\_support\_uri IN VARCHAR2, p\_token\_duration IN NUMBER, p\_refresh\_duration IN NUMBER, p\_code\_duration IN NUMBER ) RETURN ords\_types.t\_client\_key;

/\*\* \* Update an OAuth client registration.  @param p\_name The name of the client to be modified. \* This value must not be null. \* @param p\_new\_name Human readable name for the client, displayed to the end user \* during the approval phase of three-legged OAuth. \* When null, the old name is preserved. \* @param p\_description Human readable description of the purpose of the \* client, displayed to the end user during the \* approval phase of three-legged OAuth. May be null \* if p\_grant\_type == ‘client\_credentials’, non null \* otherwise. \* @param p\_privilege\_names Names of the privileges that the client wishes to \* access. Each privilege name must be separated by a comma character. \* @param p\_origins\_allowed allowed origins \* @param p\_redirect\_uri Client controlled URI to which redirect containing \* OAuth access token/error will be sent. May be null \* if p\_grant\_type == ‘client\_credentials’, non null \* otherwise. \* @param p\_support\_email Support e-mail for client’s users \* @param p\_support\_uri Support URI for client’s users \* @param p\_token\_duration Duration of the access token (bearer) in seconds. Null uses ORDS instance default value \* Which can be set by property or 3600 seconds by default. \* @param p\_refresh\_duration Duration of refresh token in seconds. \* Null uses ORDS instance default value set by a property or 86400 seconds. \* @param p\_code\_duration Duration of the code token in seconds (only applicable when grant\_type is authorization \* code. If the value is set to null or the grant\_type is not authorization\_code. The lifetime \* will be the one defined in the ords instance access\_token/12 (300 by default). \*/ PROCEDURE update\_client( p\_name IN VARCHAR2, p\_new\_name IN VARCHAR2 DEFAULT NULL, p\_description IN VARCHAR2, p\_privilege\_names IN VARCHAR2, p\_origins\_allowed IN VARCHAR2, p\_redirect\_uri IN VARCHAR2, p\_support\_email IN VARCHAR2, p\_support\_uri IN VARCHAR2, p\_token\_duration IN NUMBER, p\_refresh\_duration IN NUMBER, p\_code\_duration IN NUMBER );

### 8.2.5 Verifying an OAuth2.0 ID and secret

/\*\* \* Verifies the client identifier and secret  Returns null if the client is invalid \* Returns a cursor of rolls the client exists in the caller context and the \* id and secret are valid.  @param p\_client\_id The OAuth Client Identifier \* @param p\_client\_secret The OAuth Client Secret \* @return A cursor of the assigned roles or null is not verified. \*/ FUNCTION verify\_client( p\_client\_id IN VARCHAR2, p\_client\_secret IN VARCHAR2 DEFAULT NULL ) RETURN sys\_refcursor; END ORDS\_SECURITY;

## 8.3 OAuth2.0 Client Role and Privilege actions

### 8.3.1 Updating OAuth2.0 client privileges

/\*\* \* Updates the OAuth client privileges  @param p\_client\_key The key (id|name|client\_id) of the client to be modified. \* A minimum of one key must be supplied. \* @param p\_privilege\_names Names of the privileges that the client wishes to \* access. Each privilege name must be separated by a comma character. \*/ PROCEDURE update\_client\_privileges( p\_client\_key IN ords\_types.t\_client\_key, p\_privilege\_names IN VARCHAR2 );

/\*\* \* Updates the OAuth client privileges  @param p\_name The name of the client to be modified. \* This value must not be null. \* @param p\_privilege\_names Names of the privileges that the client wishes to \* access. Each privilege name must be separated by a comma character. \*/ PROCEDURE update\_client\_privileges( p\_name IN VARCHAR2, p\_privilege\_names IN VARCHAR2 );

### 8.3.2 Granting an OAuth2.0 client role

/\*\* \* Grant an OAuth client with the specified role.  \* @param p\_client\_key The key (id|name|client\_id) of the client to be deleted \* A minimum of one key must be supplied. \* @param p\_role\_name The name of a role that either belongs to the schema or is a built in role. \* This value must must not be null. \*/ PROCEDURE grant\_client\_role( p\_client\_key IN ords\_types.t\_client\_key, p\_role\_name IN VARCHAR2 );

/\*\* \* Grant an OAuth client with the specified role.  \* @param p\_client\_name The name of the client. \* This value must not be null \* @param p\_role\_name The name of a role that either belongs to the schema or is a built in role. \* This value must must not be null. \*/ PROCEDURE grant\_client\_role( p\_client\_name IN VARCHAR2, p\_role\_name IN VARCHAR2 );

/\*\* \* Revoke the specified role from an OAuth client, preventing \* it accessing Privileges requiring the role via two-legged OAuth  @param p\_client\_key The key (id|name|client\_id) of the client in the schema. \* A minimum of one key must be supplied. \* @param p\_role\_name The name of a role that was previously granted. \* This value must must not be null. \*/ PROCEDURE revoke\_client\_role( p\_client\_key IN ords\_types.t\_client\_key, p\_role\_name IN VARCHAR2);

/\*\* \* Revoke the specified role from an OAuth client, preventing \* it accessing Privileges requiring the role via two-legged OAuth  @param p\_client\_name The name of the client. \* This value must not be null \* @param p\_role\_name The name of a role that was previously granted. \* This value must must not be null. \*/ PROCEDURE revoke\_client\_role( p\_client\_name IN VARCHAR2, p\_role\_name IN VARCHAR2);

## 8.4 Managing OAuth2.0 Client Secrets

### 8.4.1 Generate (Rotate) a new OAuth2.0 secret by client\_key

/\*\* \* Generates a new OAuth client secret and, if required, deletes all existing client sessions.  If two client secrets are already registerd then the oldest will be overwritten. \* Any existing client secrets will also remain in effect unless revoked \* using the p\_revoke\_existing parameter.  IMPORTANT NOTES  The generated client secret will not be stored using this method and so requires the caller to \* save the returned value for future use. The view USER\_ORDS\_CLIENTS will not return the value either.  The view USER\_ORDS\_CLIENTS cannot return secrets that are not stored.  @param p\_client\_key The key (id|name|client\_id) of the client in the schema. \* A minimum of one key must be supplied. \* @param p\_revoke\_existing Revokes any exisiting secrets. (Default FALSE) \* @param p\_revoke\_sessions Deletes all existing client sessions when TRUE. (Default FALSE) \* @return The registered client secret value. This value must be saved by the caller for future reference. \*/

FUNCTION rotate\_client\_secret( p\_client\_key IN ords\_types.t\_client\_key, p\_revoke\_existing IN BOOLEAN DEFAULT FALSE, p\_revoke\_sessions IN BOOLEAN DEFAULT FALSE ) RETURN ords\_types.t\_client\_credentials;

### 8.4.2 Generate (Rotate) a new OAuth2.0 secret by client name

/\*\* \* Generates a new OAuth client secret and, if required, deletes all existing client sessions.  If two client secrets are already registerd then the oldest will be overwritten. \* Any existing client secrets will also remain in effect unless revoked \* using the p\_revoke\_existing parameter.  IMPORTANT NOTES  The generated client secret will not be stored using this method and so requires the caller to \* save the returned value for future use. The view USER\_ORDS\_CLIENTS will not return the value either.  The view USER\_ORDS\_CLIENTS cannot return secrets that are not stored.  @param p\_name The name of the client to be modified. \* This value must not be null. \* @param p\_revoke\_existing Revokes any exisiting secrets. (Default FALSE) \* @param p\_revoke\_sessions Deletes all existing client sessions when TRUE. (Default FALSE) \* @return The registered client secret value. This value must be saved by the caller for future reference. \*/

FUNCTION rotate\_client\_secret( p\_name IN VARCHAR2, p\_revoke\_existing IN BOOLEAN DEFAULT FALSE, p\_revoke\_sessions IN BOOLEAN DEFAULT FALSE ) RETURN VARCHAR2;

### 8.4.3 Revoke an OAuth2.0 client’s secret by client\_key

/\*\* \* Revoke one or both OAuth client secrets and revokes all sessions when required  By default this will only revoke the oldest secret but can be used to revoke one or both secrets \* through the use of the p\_filter parameter. The filter’s fields work independently of each other.  The special value 3 for the slot number indicated that both slots are to be revoked.  @param p\_client\_key The key (id|name|client\_id) of the client. \* A minimum of one key must be supplied. \* @param p\_filter Filter which secret(s) should be revoked. \* When the filter is null then only the oldest secret is revoked. \* When p\_filter.slot = 3 then both slots will be revoked. \* When p\_filter.stored = FALSE then this only matches when used in isolation. \* @param p\_revoke\_sessions Deletes all existing client sessions when TRUE. (Default FALSE) \* @return The client key (including client\_id) and the slot of revoked client\_secret of the client \* For the returned slot number, a value of 3 indicates that both slots were revoked and \* a null value indicated that no slots were revoked. All other client\_secret fields are null. \*/ FUNCTION revoke\_client\_secrets( p\_client\_key IN ords\_types.t\_client\_key, p\_filter IN ords\_types.t\_client\_secret DEFAULT ords\_constants.oauth\_client\_secret\_default, p\_revoke\_sessions IN BOOLEAN DEFAULT FALSE ) RETURN ords\_types.t\_client\_credentials;

### 8.4.4 Revoke an OAuth2.0 client’s secret by client name

/\*\* \* Revoke a OAuth client secret and revokes all sessions when required  By default this will only revoke the oldest secret but can be used to revoke one or both secrets \* if they match the client secret value.  @param p\_name Human readable name for the client, displayed to the end user \* during the approval phase of three-legged OAuth. \* This value must not be null \* @param p\_client\_secret The value of the client secret. When NULL, the oldest secret is revoked \* @param p\_revoke\_sessions Deletes all existing client sessions when TRUE. (Default FALSE) \*/ PROCEDURE revoke\_client\_secret( p\_name IN VARCHAR2, p\_client\_secret IN VARCHAR2 DEFAULT NULL, p\_revoke\_sessions IN BOOLEAN DEFAULT FALSE );

## 8.5 Managing OAuth2.0 client tokens

### 8.5.1 Updating access token duration for an OAuth2.0 client

#### 8.5.1.1 Updating an OAuth2.0 client’s access token duration by client\_key

/\*\* \* Updates the OAuth client token durations  @param p\_client\_key The key (id|name|client\_id) of the client to be modified. \* A minimum of one key must be supplied. \* @param p\_token\_duration Bearer duration in seconds. \* @param p\_refresh\_duration Refresh duration in seconds where applicable. \* @param p\_code\_duration Code duration in seconds. \*/ PROCEDURE update\_client\_token\_duration( p\_client\_key IN ords\_types.t\_client\_key, p\_token\_duration IN NUMBER, p\_refresh\_duration IN NUMBER, p\_code\_duration IN NUMBER );

#### 8.5.1.2 Updating an OAuth2.0 client’s access token duration by client name

/\*\* \* Updates the OAuth client token durations  @param p\_name The name of the client to be modified. \* This value must not be null. \* @param p\_token\_duration Bearer duration in seconds. \* @param p\_refresh\_duration Refresh duration in seconds where applicable. \* @param p\_code\_duration Code duration in seconds. \*/ PROCEDURE update\_client\_token\_duration( p\_name IN VARCHAR2, p\_token\_duration IN NUMBER, p\_refresh\_duration IN NUMBER, p\_code\_duration IN NUMBER );

## 8.6 Managing an OAuth2.0 client’s logo

### 8.6.1 Updating an OAuth2.0 client’s logo using the client’s key

/\*\* \* Updates the OAuth client logo file  @param p\_client\_key The key (id|name|client\_id) of the client to be modified. \* A minimum of one key must be supplied. \* @param p\_content\_type The content type of the logo \* @param p\_logo The logo binary \*/ PROCEDURE update\_client\_logo( p\_client\_key IN ords\_types.t\_client\_key, p\_content\_type IN VARCHAR2, p\_logo IN BLOB );

### 8.6.2 Updating an OAuth2.0 client’s logo using the client’s name

/\*\* \* Updates the OAuth client logo file  @param p\_name The name of the client to be modified. \* This value must not be null. \* @param p\_content\_type The content type of the logo \* @param p\_logo The logo binary \*/ PROCEDURE update\_client\_logo( p\_name IN VARCHAR2, p\_content\_type IN VARCHAR2, p\_logo IN BLOB );

1. As the ADMIN user, you can create JWT Profiles for other users (Schemas). In order to do this, you’ll need to refer to the CREATE\_JWT\_PROFILE procedure in the ORDS\_SECURITY\_ADMIN PL/SQL Package. [↑](#footnote-ref-21)
2. You can obtain the details of your OAuth2.0 client registrations with the ORDS\_METADATA.USER\_ORDS\_CLIENTS ORDS-provided view. Simply execute a SELECT \* FROM ORDS\_METADATA.USER\_ORDS\_CLIENTS; query to review the details of your client registrations. [↑](#footnote-ref-36)
3. You can obtain the details of your OAuth2.0 client registrations with the ORDS\_METADATA.USER\_ORDS\_CLIENTS ORDS-provided view. Simply execute a SELECT \* FROM ORDS\_METADATA.USER\_ORDS\_CLIENTS; query to review the details of your client registrations. [↑](#footnote-ref-43)
4. You can obtain the details of your OAuth2.0 client registrations with the ORDS\_METADATA.USER\_ORDS\_CLIENTS ORDS-provided view. Simply execute a SELECT \* FROM ORDS\_METADATA.USER\_ORDS\_CLIENTS; query to review the details of your client registrations. [↑](#footnote-ref-50)