

Tipps & Tricks: DDL Extrakt

Bereich:	PL/SQL	Erstellung:	02/2002 HA
Versionsinfo:	9.2 10.2 11.1	Letzte Überarbeitung:	05/2009 BK

 [Als PDF Downloaden!](#)

DDL Extrakt

Mit Version 9i wurde ein neues Package eingeführt, das es ermöglicht, Metadaten entweder zur Anzeige in XML oder als DDL-Befehle zu extrahieren: DBMS_METADATA. Es gibt verschiedenen Möglichkeiten zur Extraktion; stets findet sich der Befehl schließlich in einem CLOB wieder, der dann entweder direkt in SQL*Plus angezeigt oder z.B. über UTL_FILE in eine Datei geschrieben werden kann.

DDL_Extrakt in SQL

Die einfachste und einzige in SQL mögliche Form der DDL-Extraktion läuft über DBMS_METADATA.GET_DDL. An diese Funktion (mit Rückgabebetyp CLOB) müssen Typ und Name des Objekts übergeben werden. Optional kann bei Schemaobjekten noch der Owner mit angegeben werden; fehlt er, so wird im Schema des aktuellen Users gesucht. In SQL*Plus muss vorher in der Regel die Anzeige erweitert werden, damit der CLOB vollständig dargestellt wird. Wenn SPOOL eingeschaltet wird, kann so sehr schnell und einfach ein Skript erstellt werden.

GET_DDL ist auch in PL/SQL zulässig.

Beispiel in SQL:

```
SET LONG 10000
SELECT DBMS_METADATA.GET_DDL('TABLE', 'EMP'[, 'SCOTT']) FROM DUAL;
```

Beispiel in PL/SQL:

```
DECLARE
    v_lob CLOB;
BEGIN
    v_lob := DBMS_METADATA.GET_DDL('TABLE', 'EMP');
    DBMS_OUTPUT.PUT_LINE(DBMS_LOB.SUBSTR(v_lob, 254, 1));
    DBMS_LOB.FREETEMPORARY(v_lob);
END;
```

Mit GET_DDL kann nur der Befehl für ein einzelnes Objekt erzeugt werden; für Privilegien kann der DCL-Befehl nicht erzeugt werden. Der erzeugte Befehl beinhaltet auch Constraints, Storage-Klauseln, Logging-Informationen und Tablespace. Es gibt keine Möglichkeit, diese Informationen auszublenden.

DDL_Extrakt in PL/SQL

Innerhalb von PL/SQL stellt DBMS_METADATA eine ganze Reihe von Prozeduren und Funktionen zur Verfügung. Es können sowohl einzelne Objekte eines Typs als auch alle Objekte dieses Typs extrahiert werden, oder es wird ein Suchmuster angegeben. Je nach Objekttyp stehen unterschiedliche Filter zur Verfügung. Als "Objekttypen" sind auch System- und Objektprivilegien zulässig. Außerdem sind auch Einschränkungen möglich, wie z.B.:

- Storage-Klausel weglassen
- Tablespace weglassen
- Referentielle Constraints weglassen
- Andere Constraints weglassen

Minimal sind für die DDL-Form vier Funktions- bzw. Prozeduraufrufe nötig, die eine bestimmte Reihenfolge einhalten müssen:

- **OPEN** muss als erstes aufgerufen werden. Diese Funktion, an die der Objekttyp übergeben wird, gibt ein Handle vom Typ NUMBER zurück, das für die weiteren Aufrufe benötigt wird. Erlaubt sind alle gängigen Objekttypen: TABLE, VIEW, INDEX, SYNONYM, TRIGGER, FUNCTION, PROCEDURE, PACKAGE, TYPE, INDEXTYPE, OPERATOR, OUTLINE, sowie OBJECT_GRANT und SYSTEM_GRANT.
- **ADD_TRANSFORM** muss nach OPEN, aber vor FETCH aufgerufen werden. Diese Funktion gibt ein Transform Handle zurück, das benötigt wird, falls Einschränkungen gemacht werden sollen. Übergeben werden das OPEN-Handle und 'DDL'. Ohne diese Funktion kann nur die XML-Version geholt werden.
- **FETCH_DDL** und **FETCH_CLOB** holen per Default jeweils den DDL-Befehl zu einem Objekt (eine höhere Zahl kann vorher mit SET_COUNT eingestellt werden). An diese Funktionen, die u.U. mehrmals aufgerufen werden müssen, wird nur das OPEN-Handle übergeben. Rückgabetyt ist bei FETCH_DDL SYS.ku\$_ddls, eine Nested Table vom Objekt-Typ SYS.ku\$_ddl. SYS.ku\$_ddl seinerseits beinhaltet das CLOB-Attribut ddltext, in dem sich der DDL-Befehl findet. FETCH_CLOB dagegen gibt einen (temporären) CLOB zurück.
- **CLOSE** wird als letztes aufgerufen. An diese Prozedur wird nur das OPEN-Handle übergeben.

Filter (z.B. fest vorgegebener Name oder Namensmuster) können vor FETCH_xxx mit **SET_FILTER** gesetzt werden. Die oben angegebenen Einschränkungen können - ebenfalls vor dem ersten FETCH - mit **SET_TRANSFORM_PARAM** gemacht werden. Bei Schemaobjekten wird standardmäßig im aktuellen Schema gesucht.

Beispiel:

```

DECLARE
    lobby CLOB;
    handle NUMBER;
    thandle NUMBER;
    results SYS.ku$_ddls;
    result SYS.ku$_ddl;
BEGIN
    handle := DBMS_METADATA.open('TABLE');
    DBMS_METADATA.set_filter(handle, 'NAME', 'EMP');
    thandle := DBMS_METADATA.add_transform(handle, 'DDL');
    DBMS_METADATA.set_transform_param(thandle, 'SEGMENT_ATTRIBUTES',
                                     FALSE);
    results := DBMS_METADATA.fetch_ddl(handle);
    --[lobby := DBMS_METADATA.fetch_clob(handle);] -- FETCH_CLOB-Form
    IF results is null then
        DBMS_OUTPUT.put_line('Kein Ergebnis');
    ELSE
        result := results(1);
        lobby := result.ddltext;
        DBMS_OUTPUT.put_line(DBMS_LOB.substr(lobby, 254, 1));
    
```

```
END IF;  
DBMS_METADATA.close(handle);  
DBMS_LOB.freetemporary(lobby);  
END;
```

Hier wird der reine CREATE TABLE-Befehl ohne Storage-Klausel, Logging-Information und Tablespace (wurden durch SEGMENT_ATTRIBUTES = FALSE ausgeblendet) für die Tabelle EMP im aktuellen Schema extrahiert.

Weiteres Beispiel:

```
DECLARE  
    lobby CLOB;  
    handle NUMBER;  
    thandle NUMBER;  
BEGIN  
    handle := DBMS_METADATA.open('SYSTEM_GRANT');  
    thandle := DBMS_METADATA.add_transform(handle, 'DDL');  
    DBMS_METADATA.set_filter(handle, 'GRANTEE', 'SCOTT');  
    LOOP  
        lobby := DBMS_METADATA.fetch_clob(handle);  
        EXIT WHEN lobby IS NULL;  
        DBMS_OUTPUT.put_line(DBMS_LOB.substr(lobby, 254, 1));  
    END LOOP;  
    DBMS_METADATA.close(handle);  
END;
```

Hier werden alle Systemprivilegien extrahiert, die SCOTT erhalten hat.

Die Prozeduren/Funktionen sind case-sensitive. Sie müssen also Typ, Name und Owner stets mit Großbuchstaben übergeben. Objekte, auf die ein User keine Zugriffsrechte hat, kann er auch nicht extrahieren.

Weitere Einzelheiten finden Sie in der Oracle-Dokumentation zu DBMS_METADATA.