

# KAPITEL 2

## 2 Einführung in die Oracle-Datenbank

*Oracle, lateinisch oraculum* [von oro = ich spreche, ich bete]

Es gibt drei wesentlich unterschiedliche Kategorien von Benutzern einer Oracle-Datenbank:

- ✓ Systemverwalter
- ✓ Entwickler
- ✓ Endanwender

Das vorliegende Buch zielt auf die zweite Benutzergruppe ab. Sie muss nicht in der Lage sein, Aufgaben des Systemverwalters wahrzunehmen. Es ist dennoch sehr nützlich, wenn man versteht, wie die einzelnen Objekte einer Oracle-Datenbank zusammenspielen, beziehungsweise wenn man eine hinreichend tragfähige Vorstellung von diesem Zusammenspiel hat. Eine solche Modellvorstellung zu vermitteln, das ist Anliegen des Kapitels.

Die ersten Kenntnisse über das Oracle-Datenbanksystem gewinnt man am besten über SQL\*Plus. Vom Betriebssystem aus startet man meist mit dem Befehl:

```
sqlplus Benutzer/Passwort
```

Genauer über die Syntax des Zeilenkommandos sqlplus erfährt man über den Aufruf

```
sqlplus -c
Usage: SQLPLUS [<option>] [<user>[/<password>] [@<host>]]
           [@<startfile> [<parm1> [<parm2>] ...]
where <option> ::= { -s | -? }
-s for silent mode and -? to obtain version number
```

## 2.1 Tabellen

Eine Oracle-Datenbank ist eine Sammlung von Tabellen. (Neben den Tabellen gibt es natürlich noch eine Fülle anderer Objekte, die eine Oracle-Datenbank ausmachen.) In den Tabellen werden die Daten einer Applikation gespeichert. Es ist daher nötig, sich einen genaueren Begriff vom Wesen einer Tabelle zu machen.

Eine Tabelle hat einen Namen. Sie besteht aus Spalten und Zeilen. Den Spalten ist neben einem Namen ein Datentyp zugeordnet. Tabellen werden mittels des SQL-Befehls **CREATE** eingerichtet. Mit dem SQL-Befehl **SELECT** werden Inhalte von Tabellen sichtbar gemacht. Das folgende Beispiel legt eine Tabelle mit dem Namen HILF und den Spalten F1, F2, F3 und F4 an.

```
create table HILF
(
  F1number,
  F2date,
  F3char(30),
  F4number(9,2) );
```

Abbildung 4: CREATE einer Tabelle

Diese Tabelle besteht aus den vier Spalten F1 bis F4. Jede Spalte hat einen eigenen Datentyp, der aus Abbildung 4 ersichtlich ist. Die Tabelle wird mit dem SQL-Befehl **CREATE** eingerichtet, das heißt als Objekt von der Art »TABLE« in der Datenbank definiert.

Mit dem SQL-Befehl **INSERT** kann man Daten in die Tabelle einfügen, wie etwa aus folgender Befehlssequenz hervorgeht:

```
insert into HILF values (1, SYSDATE, '1 kg Bananen', 5.20 );
insert into HILF values (2, SYSDATE, '2 kg Birnen' , 3.00 );
insert into HILF values (3, SYSDATE, '1 kg Feigen' ,12.30 );
insert into HILF values (4, SYSDATE, '1 kg Gurken' , 2.40 );
insert into HILF values (5, SYSDATE, '1 kg Orangen', 2.60 );
```

Abbildung 5: Einfügen von Zeilen in eine Tabelle

Offensichtlich werden durch diese Befehlsfolge fünf Zeilen (= Sätze) in die Tabelle eingefügt. Dabei wird den vier Spalten je ein gültiger Wert zugeordnet. **SYSDATE** ist dabei eine Standardfunktion, die das aktuelle Tagesdatum und die Uhrzeit in der Spalte F2 ablegt.

Nun kann man den Inhalt der Tabelle HILF wieder sichtbar machen. Dazu dient der Befehl **SELECT**. Die einfachste Anwendung dieses Befehls hat die Form:

```
select * from HILF;
```

Hiermit werden alle Spalten und alle Zeilen der Tabelle HILF sichtbar gemacht, wie aus Abbildung 6 hervorgeht.

F1	F2	F3	F4
1	15.04.00	1 kg Bananen	5,2
2	15.04.00	2 kg Birnen	3
3	15.04.00	1 kg Feigen	12,3
4	15.04.00	1 kg Gurken	2,4
5	15.04.00	1 kg Orangen	2,6

Abbildung 6: Anwendung des Befehls SELECT

Die Struktur einer als Objekt in der Datenbank abgelegten Tabelle kann jederzeit über den Befehl **DESC** (= **describe**) wieder sichtbar gemacht werden. Wir tippen ein:

```
desc hilf
```

und erhalten ein Resultat wie in Abbildung 7.

Name	Null?	Type
F1		NUMBER
F2		DATE
F3		VARCHAR2(30)
F4		NUMBER(9,2)

Abbildung 7: Anwendung des Befehls desc auf die Tabelle HILF

Daten **über** Datenbankobjekte wie Tabellen, Benutzer und andere Objekte, die wir noch kennen lernen werden, werden als **Metadaten** bezeichnet. Diese werden wie normale »Nutzdaten« in Tabellen gespeichert. Man nennt derartige Tabellen oft auch *Systemtabellen*, *Data Base Views*, *Datenbanksichten* oder eine *Metadatenbank*. Die wichtigsten Vertreter hiervon sind die Objekte **TAB** und **CAT**, die für jede Tabelle, die kreiert worden ist, einen Eintrag anzeigen. Ihre Struktur, mit dem Befehl desc sichtbar gemacht, ist in Abbildung 8 enthalten.

```
SQL> desc tab
Name                Null?   Type
TNAME               NOT NULL VARCHAR2(30)
TABTYPE              VARCHAR2(7)
CLUSTERID            NUMBER
SQL> desc cat
Name                Null?   Type
TABLE_NAME          NOT NULL VARCHAR2(30)
TABLE_TYPE           VARCHAR2(11)
```

Abbildung 8: Die Strukturen von TAB und CAT

Den Inhalt der Datenbanksichten TAB oder CAT kann man mit dem SQL-Befehl

```
select * from TAB;  
select * from cat;
```

sichtbar machen. Eine ausführliche Erklärung des SELECT-Befehls folgt in einem späteren Abschnitt.

Da Benutzer, Rollen, Privilegien, Constraints, Datenbanktrigger und viele andere Metadaten in entsprechenden Systemtabellen geführt werden, kann man sich durch die Kenntnis dieser Tabellen viele Informationen über eine Oracle-Datenbank erarbeiten. Die eben aufgezählten Objekte sind Beispiele von Dingen, die vom Systemverwalter oder von Entwicklern in einer Oracle-Datenbank definiert werden.

**Syntaktische Anmerkung:** Groß- und Kleinschreibung spielt in SQL keine Rolle. Die Namen von Tabellen und Spalten dürfen maximal 30 Zeichen lang sein. Eine SQL-Anweisung endet mit einem Semikolon.

## 2.2 Datenbanken physisch und logisch

Das Datenbanksystem Oracle ist auf sehr vielen Betriebssystemen installierbar. Aus der Sicht des jeweiligen Betriebssystems umfasst eine Oracle-Datenbank vier verschiedene Kategorien von Dateien. Dabei wird unter »Datei« das verstanden, was das jeweilige Gastbetriebssystem darunter versteht. Eine Oracle-Datenbank besteht aus folgenden Dateien:

- ✓ beliebig viele Datenbankdateien
- ✓ mindestens zwei REDO-Logdateien
- ✓ mindestens zwei Kontrolldateien
- ✓ eine Parameterdatei

Die Datenbankdateien können auf beliebige Plattenlaufwerke verteilt sein. Sie enthalten die eigentliche Oracle-Datenbank beziehungsweise die Oracle-Datenbanken. Hierin lagern die Datenbankobjekte wie Tabellen, Views etc.

Die REDO-Logdateien enthalten Log-Information der Transaktionen bei Änderung der Datenbank. Sie dienen auch zur Wiederherstellung der Datenbank bei einem Absturz oder sonstigen Störungen (Recovery).

Die Kontrolldateien enthalten Informationen über die Grundstrukturen und Grundinformationen der Datenbank, zum Beispiel:

- ✓ der Zeitpunkt der Datenbank-Erstellung
- ✓ die Namen der Datenbankdateien
- ✓ die Namen der REDO-Logdateien
- ✓ Zeitpunkt des letzten Checkpoints

Eine Datenbank wird mit dem SQL-Befehl CREATE DATABASE eingerichtet:

```
create database ENERGIE
controlfile reuse
logfile group 1 ('C:LOG1.LOG, D:LOG1.LOG') size 100K,
logfile group 2 ('C:LOG2.LOG, D:LOG2.LOG') size 100K,
maxlogfiles 5
maxloghistory 100
datafile 'C:ENERGIE.DAT' size 2M
maxdatafiles 10
maxinstances 2
archivelog
exclusive
```

Hierdurch wird eine Datenbank mit dem Namen ENERGIE eingerichtet. Die zugehörige physische Datei liegt auf Platte C und hat den Namen ENERGIE.DAT. Ferner sind zwei REDO-Logdateien definiert. Kontrolldateien werden durch einen separaten CREATE-Befehl eingerichtet:

```
create controlfile
```

Mit dieser physischen Struktur der Datenbank wird nur der Datenbankadministrator (DBA) konfrontiert. Da die genaue Darstellung aller Einzelheiten sehr komplex ist und die Zielsetzung dieses Buches übersteigt, wird auf weitere Erklärungen verzichtet. Die Datenbankdateien können über eine Systemtabelle mit dem (Synonym-) Namen DBA\_DATA\_FILES angezeigt werden. Eine Beispielaufstellung findet sich in Abbildung 9. Dabei ist unter der Spalte FILE\_NAME der Name der physischen Datei im Betriebssystem (in diesem Falle Unix).

FILE_NAME	DBA_DATA_FILES		
	FILE_ID	TABLESPACE	BYTES
/Oracle7/dbs/systora7.dbf	1	SYSTEM	26214400
/Oracle7/dbs/rbsora7.dbf	2	RBS	26214400
/Oracle7/dbs/tempora7.dbf	3	TEMP	26214400
/Oracle7/dbs/toolora7.dbf	4	TOOLS	26214400
/Oracle7/dbs/usrora7.dbf	5	USERS	52428800
/Oracle7/dbs/testora7.dbf	6	TESTSPACE	53245
/Oracle7/dbs/test2ora7.dbf	7	TESTSPACE	53245
/Oracle7/dbs/usr2ora7.dbf	8	USERS	62914560
/Oracle7/dbs/rbs2ora7.dbf	9	RBS	10455760
/usr/Oracle7/rbs3ora7.dbf	10	RBS	10455760
/usr/Oracle7/rbs4ora7.dbf	11	RBS	20971520
/usr/Oracle7/usr3ora7.dbf	12	USERS	41943040
/usr/Oracle7/usr4ora7.dbf	13	USERS	20971520
/ora7_dbf/rbs5ora7.dbf	14	RBS	31457280
/ora7_dbf/usr5ora7.dbf	15	USERS	20971520

Abbildung 9: Beispielaufstellung aus DBA\_DATA\_FILES

Die Datenbankdateien, die sich auf einem Rechnersystem befinden, werden logisch in so genannte Tablespaces unterteilt. Ein Tablespace ist eine Zuordnung von Speicherplatz innerhalb einer Datenbank. Eine Oracle-Datenbank besteht aus Tablespaces. Die Objekte der Datenbank sind über Tablespaces abgegrenzt. Tablespaces sind folgendermaßen gegliedert:

- ✓ Tabellen (Datensegmente)
- ✓ Indexe (Indexsegmente)
- ✓ Rollback-Segmente
- ✓ Temporäre Segmente

Ein Tablespace wird ebenfalls durch einen CREATE-Befehl eingerichtet. Alle eingerichteten Tablespaces sind über das Synonym DBA\_TABLESPACES erreichbar. (Mit dem Befehl CREATE SYNONYM kann ein Synonym zu einer Tabelle angelegt werden, über das dann die betreffende Tabelle auch angesprochen werden kann. Synonyme dienen dazu, die Namensvergabe von Objekten mnemotechnisch neuen Umgebungen anzupassen.) Ein Auszug aus dieser Tabelle könnte ein Bild wie in Abbildung 10 ergeben.

Tablespace	Ini-Ext	Nxt-Ext	Min.E.	Max.E	%-Incr	STATUS
SYSTEM	12288	12288	1	249	50	ONLINE
RBS	131072	131072	2	249		ONLINE
TEMP	262144	262144	1	249		ONLINE
TOOLS	20450	20450	1	249	50	ONLINE
USERS	20450	20450	1	249	50	ONLINE
TESTSPACE	12288	12288	1	20		ONLINE

Abbildung 10: Beispielausgabe für DBA\_TABLESPACES

Das Einrichten von Tabellen beansprucht Speicherplatz in einem Tablespace. Diesen fordert man über eine Speicherklausel an. Fehlt sie beim CREATE einer Tabelle, so werden Standardwerte verwendet. Zum professionellen Umgang mit SQL gehört jedoch stets eine explizite und wohl überlegte Angabe der Speicherklausel.

#### Die Speicherklausel (Storage Clause)

```
storage
( initial 20MB
  next 10MB
  minextents 3
  maxextents 80
  pctincrease 0
  freelists 3)
```

ist in folgenden SQL-Befehlen anwendbar:

CREATE/ALTER	Tablespace
CREATE/ALTER	Table
CREATE/ALTER	Index
CREATE/ALTER	Rollback Segment
CREATE/ALTER	Cluster
CREATE/ALTER	Snapshot
CREATE/ALTER	Snapshot Log

Beispiel einer Anwendung der Speicherklausel:

```
create tablespace <NAME_TABSPACE>
datafile 'DISCA: DATEI' size 20M
default storage
(initial 10K
 next 50K
 minextents 1
 maxextents 999
 pctincrease 10 )
online
```

Abbildung 11: Beispiel für Create Tablespace

Weitere Einzelheiten über das Kommando CREATE TABLESPACE werden an dieser Stelle nicht erwähnt.

Einige weitere nützliche Datenbanksichten sind (siehe auch Abbildung 21):

DBA_TABLESPACES	Liste der Tablespaces
DBA_DATA_FILES	Liste der Datenbankdateien
USER/DBA_EXTENTS	Liste der Extents je Segmenttyp
USER/DBA_FREE_SPACE	Freie Blöcke je Segment
USER/DBA_SEGMENTS	Liste der Segmente

Eine weitere Möglichkeit, einen Tablespace zu überwachen, besteht darin, die Tabelle DBA\_FREE\_SPACE zu betrachten. Ein Auszug aus dieser Tabelle könnte folgendes Bild ergeben:

Tablespace	FILE_ID	BLOCK_ID	BYTES	BLOCKS
SYSTEM	1	998	11458176	356
SYSTEM	1	4582	679936	166
SYSTEM	1	1447	8192	2
SYSTEM	1	2495	1097728	268
SYSTEM	1	3316	880640	215
SYSTEM	1	2061	851968	208
RBS	2	6372	118784	29
RBS	11	5112	36864	9
RBS	14	2	31453184	7679
TEMP	3	2	26210304	6399
TOOLS	4	2562	143360	35
TOOLS	4	2692	15192064	3709
USERS	5	411	1945600	475
USERS	8	394	20779008	5073

Abbildung 12: Beispielauszug aus DBA\_FREE\_SPACE

## 2.3 Zutritt zur Datenbank

### 2.3.1 Benutzer


Beim Thema Datenbankzutritt sind naturgemäß zunächst die Benutzer der Datenbank zu erwähnen. Diese werden vom Systemverwalter erfasst. Sie sind (wenn man die entsprechende Berechtigung hat) aus DBA\_USERS zu ermitteln:

SQL> desc dba_users				
Name	Null?	Type		
-----	-----	----		
USERNAME	NOT NULL	VARCHAR2(30)		
USER_ID	NOT NULL	NUMBER		
PAssWORD		VARCHAR2(30)		
ACCOUNT_STATUS	NOT NULL	VARCHAR2(32)		*)
LOCK_DATE		DATE		*)
EXPIRY_DATE		DATE		*)
DEFAULT_TABLESPACE	NOT NULL	VARCHAR2(30)		
TEMPORARY_TABLESPACE	NOT NULL	VARCHAR2(30)		
CREATED	NOT NULL	DATE		
PROFILE	NOT NULL	VARCHAR2(30)		
EXTERNAL_NAME		VARCHAR2(4000)		*)

Abbildung 13: Struktur von DBA\_USERS

Aus dieser Struktur sind viele Einzelheiten ablesbar. So sieht man bereits, welche Information von einem Systemverwalter eingegeben werden muss, damit ein Benutzer in Oracle definiert werden kann. Dabei sind die ersten drei Datenfelder USERNAME, USER\_ID und PASSWORD offensichtlich zu verstehen. Das Feld CREATED ist die Zeit, zu der ein Benutzer angelegt worden ist.



 **Oracle8** Die in Abbildung 13 mit \*) markierten Spalten sind neu in Oracle8. Man kann also die Zutrittskontrolle eines Benutzers über zwei Datumsangaben regeln, ein Sperrdatum (LOCK\_DATE) und ein Verfallsdatum (EXPIRY\_DATE). Daraus resultiert zwangsläufig, dass man eine Statusinformation benötigt (ACCOUNT\_STATUS). Der normale Status ist OPEN. Die Spalte EXTERNAL\_NAME erlaubt es, einen beschreibenden Text über den Benutzer zu speichern.

Jedem Benutzer wird außerdem per Default ein Tablespace zugeordnet, auf dem er operieren kann. Ferner benötigt ein Benutzer einen zweiten temporären Tablespace für temporäres Auslagern von Daten, wie dies beispielsweise bei Sortiervorgängen immer nötig ist. Außerdem braucht ein Benutzer ein PROFILE. (Dieser Begriff wird weiter unten erklärt werden.)

Weiterhin sind für einen Benutzer Speicherplatzberechtigungen notwendig. Diese sind aus DBA\_TS\_QUOTAS abzulesen, deren Struktur in Abbildung 14 enthalten ist.

```
SQL> desc dba_ts_quotas
Name                                Null?    Type
-----
TABLESPACE_NAME                     NOT NULL VARCHAR2(30)
USERNAME                             NOT NULL VARCHAR2(30)
BYTES                               NUMBER
MAX_BYTES                           NUMBER
BLOCKS                              NOT NULL NUMBER
MAX_BLOCKS                           NUMBER
```

Abbildung 14: Struktur von DBA\_TS\_QUOTAS

### 2.3.2 Profile

Ein Profile ist eine Menge von (gewissen) Restriktionen, die unter einem Namen als Objekt in der Datenbank abgelegt werden können. Ein definiertes Profile kann Benutzern zugewiesen werden. Dabei hat ein Benutzer immer nur ein Profile. Ein Profile wird mit dem Befehl

```
create profile
```

angelegt. Einige der Parameter, die dabei einzustellen sind, lauten:

- ✓ SESSIONS\_PER\_USER      Anzahl gleichzeitiger Dialoganmeldungen
- ✓ CPU\_PER\_SESSION        Schranke für CPU-Zeitverbrauch je Sitzung (in sec/100)
- ✓ IDLE\_TIME                Schranke für Abschalten der Anmeldung bei Nichtbenutzung des Bildschirms

Die definierten Profiles sind über das Synonym DBA\_PROFILES zu finden.

```
SQL> desc dba_profiles
Name                                Null?    Type
-----
PROFILE                             NOT NULL VARCHAR2(30)
RESOURCE_NAME                       NOT NULL VARCHAR2(32)
RESOURCE_TYPE                        VARCHAR2(8)
LIMIT                               VARCHAR2(40)
```

Abbildung 15: Struktur von DBA\_PROFILES

### 2.3.3 Systemprivilegien

Oracle kennt eine feste, wohl definierte Liste von Systemprivilegien. Diese sind im Handbuch »SQL Language Reference Manual« definiert und aufgelistet. Jedes Systemprivileg hat eine eindeutige Bezeichnung, zum Beispiel

✓ **ALTER DATABASE**      Recht, Datenbankdefinition zu ändern

Die Systemprivilegien sind über das Synonym DBA\_SYS\_PRIVS erreichbar, dessen Struktur in Abbildung 16 zu sehen ist.

```
SQL> desc dba_sys_privs
Name                                Null?    Type
-----
GRANTEE                             NOT NULL VARCHAR2(30)
PRIVILEGE                           NOT NULL VARCHAR2(40)
ADMIN_OPTION                        VARCHAR2(3)
```

Abbildung 16: Struktur von DBA\_SYS\_PRIVS

In Oracle gibt es ca. 90 verschiedene Systemprivilegien, die in Anhang C enthalten sind.

### 2.3.4 Objektprivilegien

Für jedes Objekt, zum Beispiel eine Tabelle, kann es folgende Objektprivilegien geben:

<b>ALTER</b>	Objekt ändern
<b>DELETE</b>	Objekt löschen
<b>EXECUTE</b>	Funktion oder Prozedur ausführen
<b>INDEX</b>	Index auf Tabelle anlegen
<b>INSERT</b>	Einfügen auf Tabelle
<b>REFERENCES</b>	Constraints mit Referenz auf Tabelle anlegen
<b>SELECT</b>	Tabelle lesen
<b>UPDATE</b>	Tabelle verändern

System- und Objektprivilegien können an Benutzer oder Rollen vergeben werden. Hierzu gibt es den SQL-Befehl **GRANT** beziehungsweise dessen Gegenteil **REVOKE**.

### 2.3.5 Rollen

Eine Rolle ist eine Menge von System- oder Objektprivilegien. Mittels des SQL-Befehls

```
create role
```

wird eine Rolle angelegt, indem ein Name der Rolle angegeben wird. Die Rollen werden in der Datenbanksicht `DBA_ROLES` geführt (Abbildung 17).

```
SQL> desc dba_roles
Name                                Null?    Type
-----
ROLE                                NOT NULL VARCHAR2(30)
PASSWORD_REQUIRED                   VARCHAR2(8)
```

Abbildung 17: Struktur von `DBA_ROLES`

Oracle kennt folgende fünf Standardrollen:

- ✓ CONNECT
- ✓ RESOURCE
- ✓ DBA
- ✓ EXP\_FULL\_DATABASE
- ✓ IMP\_FULL\_DATABASE

Einer Rolle können mehrere Privilegien zugeordnet werden. Dazu gibt es die Datenbanksicht `DBA_ROLE_PRIVS` (Abbildung 18).

```
SQL> desc dba_role_privs
Name                                Null?    Type
-----
GRANTEE                             VARCHAR2(30)
GRANTED_ROLE                         NOT NULL VARCHAR2(30)
ADMIN_OPTION                         VARCHAR2(3)
DEFAULT_ROLE                         VARCHAR2(3)
```

Abbildung 18: Struktur von `DBA_ROLE_PRIVS`

Rollen für Objektprivilegien auf Tabellen werden in einer gesonderten Datenbanksicht geführt (Abbildung 19).

```
SQL> desc dba_tab_privs
Name                                Null?    Type
-----
GRANTEE                            NOT NULL VARCHAR2(30)
OWNER                              NOT NULL VARCHAR2(30)
TABLE_NAME                         NOT NULL VARCHAR2(30)
GRANTOR                            NOT NULL VARCHAR2(30)
PRIVILEGE                          NOT NULL VARCHAR2(40)
GRANTABLE                           VARCHAR2(39)
```

Abbildung 19: Struktur von ROLE\_TAB\_PRIVS

## 2.4 Datenbankobjekte

Datenbankobjekte sind über DBA\_OBJECTS zugänglich (Abbildung 20).

```

Name                                Null?    Type
-----
OWNER                              VARCHAR2(30)
OBJECT_NAME                       VARCHAR2(128)
SUBOBJECT_NAME                    VARCHAR2(30)
OBJECT_ID                         NUMBER
DATA_OBJECT_ID                   NUMBER
OBJECT_TYPE                       VARCHAR2(15)
CREATED                          DATE
LAST_DDL_TIME                    DATE
TIMESTAMP                        VARCHAR2(19)
STATUS                           VARCHAR2(7)
TEMPORARY                        VARCHAR2(1)
GENERATED                        VARCHAR2(1)
```

Abbildung 20: Struktur von DBA\_OBJECTS

Durch den Befehl CREATE werden Datenbankobjekte angelegt. Es gibt folgende Objekte, die für den programmierenden Anwender von Oracle eine Rolle spielen und die in diesem Buch bei der Diskussion von SQL behandelt werden:

- ✓ FUNCTION
- ✓ INDEX
- ✓ PACKAGE
- ✓ PACKAGE BODY
- ✓ PROCEDURE
- ✓ SEQUENCE
- ✓ SYNONYM
- ✓ TABLE

✓ TRIGGER

✓ VIEW

☞ **Oracle8** Die Typen der Datenbankobjekte sind in Oracle8 gegenüber den früheren Versionen erheblich erweitert worden. Eine Liste von weiteren Objekttypen ist folgende:

✓ CLUSTER (schon in früheren Versionen vorhanden)

✓ DIRECTORY

✓ LIBRARY

✓ TYPE

✓ TYPE BODY

✓ UNDEFINED

Abbildung 21 enthält eine Aufzeichnung einiger Tabellen (Synonyme) der Metadatenbank. Über die Datenbanksicht DICT findet man alle Tabellen der Metadatenbank:

SQL> desc dict		
Name	Null?	Type
-----	-----	-----
TABLE_NAME		VARCHAR2(30)
COMMENTS		VARCHAR2(4000)

DICT repräsentiert also den Zugang zum »Data Dictionary« von Oracle. Mit dem SELECT-Befehl

```
select table_name, comment from dict
where table_name like 'DBA_%';
```

kann man alle Synonyme des DBA sichtbar machen. Der erste Teil der Abbildung 21 zeigt das Resultat dieser Abfrage.

DBA_2PC_NEIGHBORS	information about incoming and outgoing connections for pending transact
DBA_2PC_PENDING	info about distributed transactions awaiting recovery
DBA_ANALYZE_OBJECTS	
DBA_AUDIT_EXISTS	Lists audit trail entries produced by AUDIT NOT EXISTS
	and AUDIT EXISTS
DBA_AUDIT_OBJECT	Audit trail records for statements concerning objects, specifically: table, cluster, view, index, sequence, [public] database link, [public] synonym, procedure, trigger, rollback segment, tablespace, role, user
DBA_AUDIT_SESSION	All audit trail records concerning CONNECT and DISCONNECT
DBA_AUDIT_STATEMENT	Audit trail records concerning grant, revoke, audit, noaudit and alter system
DBA_AUDIT_TRAIL	All audit trail entries
DBA_CATALOG	All database Tables, Views, Synonyms, Sequences
DBA_CLUSTERS	Description of all clusters in the database
DBA_CLUSTER_HASH_EXPRESSIONS	Hash functions for all clusters
DBA_CLU_COLUMNS	Mapping of table columns to cluster columns
DBA_COL_COMMENTS	Comments on columns of all tables and views
DBA_COL_PRIVS	All grants on columns in the database
DBA_CONSTRAINTS	Constraint definitions on all tables
DBA_CONS_COLUMNS	Information about accessible columns in constraint definitions
DBA_DATA_FILES	Information about database files
DBA_DB_LINKS	All database links in the database
DBA_DEPENDENCIES	Dependencies to and from objects
DBA_ERRORS	Current errors on all stored objects in the database
DBA_EXP_FILES	Description of export files
DBA_EXP_OBJECTS	Objects that have been incrementally exported
DBA_EXP_VERSION	Version number of the last export session
DBA_EXTENTS	Extents comprising all segments in the database
DBA_FREE_SPACE	Free extents in all tablespaces
DBA_FREE_SPACE_COALESCED	Statistics on Coalesced Space in Tablespaces
DBA_FREE_SPACE_COALESCED_TMP1	Coalesced Free Extents for all Tablespaces
DBA_FREE_SPACE_COALESCED_TMP2	Free Extents in Tablespaces
DBA_HISTOGRAMS	Histograms on columns of all tables
DBA_INDEXES	Description for all indexes in the database
DBA_IND_COLUMNS	COLUMNS comprising INDEXes on all TABLEs and CLUSTERs
DBA_JOBS	All jobs in the database
DBA_JOBS_RUNNING	All jobs in the database which are currently running, join v\$lock and job\$
DBA_OBJECTS	All objects in the database
DBA_OBJECT_SIZE	Sizes, in bytes, of various pl/sql objects
DBA_OBJ_AUDIT_OPTS	Auditing options for all tables and views
DBA_PRIV_AUDIT_OPTS	Describes current system privileges being audited across the system and by user

Abbildung 21: Auszug aus der Metadatenbank von Oracle (Version 7.3)

DBA_PROFILES	Display all profiles and their limits
DBA_RCHILD	All the children in any refresh group. This view is not a join.
DBA_REFRESH	All the refresh groups
DBA_REFRESH_CHILDREN	All the objects in refresh groups
DBA_REPAUDIT_ATTRIBUTE	Information about attributes automatically maintained for replication
DBA_REPAUDIT_COLUMN	Information about columns in all shadow tables for all replicated tables in the database
DBA_REPCAT	
DBA_REPCATALOG	Information about asynchronous administration requests
DBA_REPCOLUMN_GROUP	All column groups of replicated tables in the database
DBA_REPCONFLICT	All conflicts for which users have specified resolutions in the database
DBA_REPDDL	Arguments that do not fit in a single repcat log record
DBA_REPGENERATED	Objects generated to support replication
DBA_REPGROUP	Information about all replicated object groups
DBA_REPGROUPED_COLUMN	Columns in the all column groups of replicated tables in the database
DBA_REPKEY_COLUMNS	Primary columns for a table using column-level replication
DBA_REPOBJECT	Information about replicated objects
DBA_REPPARAMETER_COLUMN	All columns used for resolving conflicts in the database
DBA_REPPRIORITY	Values and their corresponding priorities in all priority groups in the database
DBA_REPPRIORITY_GROUP	Information about all priority groups in the database
DBA_REPPROP	Propagation information about replicated objects
DBA_REPRESOLUTION	Description of all conflict resolutions in the database
DBA_REPRESOLUTION_METHOD	All conflict resolution methods in the database
DBA_REPRESOLUTION_STATISTICS	Statistics for conflict resolutions for all replicated tables in the data base
DBA_REPRESOL_STATS_CONTROL	Information about statistics collection for conflict resolutions for all replicated tables in the database
DBA_REPSHEMA	N-way replication information
DBA_REPSITES	N-way replication information
DBA_RGROUP	All refresh groups. This view is not a join.
DBA_ROLES	All Roles which exist in the database
DBA_ROLE_PRIVS	Roles granted to users and roles
DBA_ROLLBACK_SEGS	Description of rollback segments
DBA_SEGMENTS	Storage allocated for all database segments
DBA_SEQUENCES	Description of all SEQUENCES in the database
DBA_SNAPSHOTS	All snapshots in the database
DBA_SNAPSHOT_LOGS	All snapshot logs in the database

Abbildung 21: Auszug aus der Metadatenbank von Oracle (Version 7.3)

DBA_SOURCE	Source of all stored objects in the database
DBA_STMT_AUDIT_OPTS	Describes current system auditing options across the system and by user
DBA_SYNONYMS	All synonyms in the database
DBA_SYS_PRIVS	System privileges granted to users and roles
DBA_TABLES	Description of all tables in the database
DBA_TABLESPACES	Description of all tablespaces
DBA_TAB_COLUMNS	Columns of user's tables, views and clusters
DBA_TAB_COMMENTS	Comments on all tables and views in the database
DBA_TAB_PRIVS	All grants on objects in the database
DBA_TRIGGERS	All triggers in the database
DBA_TRIGGER_COLS	Column usage in all triggers
DBA_TS_QUOTAS	Tablespace quotas for all users
DBA_UPDATABLE_COLUMNS	Description of dba updatable columns
DBA_USERS	Information about all users of the database
DBA_VIEWS	Text of all views in the database
86 rows selected.	

Abbildung 21: Auszug aus der Metadatenbank von Oracle (Version 7.3)



**Oracle8** Dieser Auszug ist in Oracle8 etwas länger. Außerdem gibt es hier die Datenbanksicht SYSCATALOG:

Name	Null?	Type
-----	-----	-----
TNAME	NOT NULL	VARCHAR2(30)
CREATOR	NOT NULL	VARCHAR2(30)
TABLETYPE		VARCHAR2(8)
REMARKS		VARCHAR2(4000)