

# Database Systems II B: DBMS-Implementation

---

## Chapter 2: The System Catalog (Data Dictionary)

Prof. Dr. Stefan Brass

Martin-Luther-Universität Halle-Wittenberg

Wintersemester 2021/22

<http://www.informatik.uni-halle.de/~brass/dbi21/>

# Objectives

After completing this chapter, you should be able to:

- name applications and advantages of a system catalog.

Why is it good that one can use SQL to query meta-data?

- explain the structure of the Oracle Data Dictionary.
- formulate queries against data dictionary tables.
- name at least three tables of the Oracle data dictionary and their most important columns.

# Contents

- 1 Introduction
- 2 The Oracle Data Dictionary
- 3 DB Schema Information in Oracle

# Data Dictionaries (1)

- Most DBMS have a large collection of system tables, called the “data dictionary” (system catalog).
- In Oracle, it contains the following information:
  - Tables, views, etc. (database schema).
  - Comments on tables/columns (documentation).
  - Database Users, Access Rights, Auditing Data.
  - Indexes, Physical Storage Parameters, File space usage (e.g. allocation of disk blocks for tables).
  - Statistical Information, Performance Data.

# Data Dictionaries (2)

- Data dictionaries are very system-dependent.

The tables Oracle uses are completely different from those used in DB2 or SQL Server. The data dictionary even changed substantially between different versions of Oracle. However, the SQL-92 standard proposes an information schema with some standard views (currently only implemented in SQL Server).

- The data dictionary is an important tool for the DBA!

All information that the DBMS has should be available in system tables. Therefore, to understand all information in the data dictionary is to understand the system. A seasoned DBA should know many of the data dictionary tables (at least 50). The Oracle certification exams also contain exercises that ask for data dictionary tables.

# Data Dictionaries (3)

- Of course, modern DBMS also have a graphical user interface to the system data, e.g. the Oracle Enterprise Manager.

Or: <http://dbs.informatik.uni-hannover.de/ftp/software/oddis/>

- However, such tools fetch their information from the data dictionary.
- In addition, they support only browsing of the data. More complicated evaluations still must be done using SQL queries (possibly in scripts) to the data dictionary. E.g. find disks that are nearly full.

# Contents

- 1 Introduction
- 2 The Oracle Data Dictionary
- 3 DB Schema Information in Oracle

# Example: Catalog (1)

- Names of schema objects (e.g. tables, columns) are now stored as data in the database.
- Such data is called “meta-data” (data about data).
- In this way, queries to data and meta-data can be formalized in the same language.

A general query language like SQL is much more powerful than a specialized set of commands like “**describe**” in Oracle SQL\*Plus.

- E.g., this query lists all tables of the current user:

```
SELECT TABLE_NAME FROM CAT
```

- **CAT** is a table (view) from the data dictionary.



# Example: Catalog (2)

- E.g., for the guest user SCOTT, CAT looks as follows:

CAT	
TABLE_NAME	TABLE_TYPE
DEPT	TABLE
EMP	TABLE
⋮	⋮

- CAT lists also views, sequences, synonyms.

CAT lists all table-like objects (TABLE\_TYPE shows the exact type). CAT is not listed because it is not owned by SCOTT. Sequences are generators for unique numbers, for synonyms see below. Both exist only in Oracle.

- Note that table names etc. are stored in uppercase!

While normally, case is not important for table and column names, it is important for string data.

# Example: Catalog (3)

- All table-like objects accessible by the current user are listed in the data dictionary view “**ALL\_CATALOG**”:

ALL_CATALOG		
OWNER	TABLE_NAME	TABLE_TYPE
SCOTT	BONUS	TABLE
SCOTT	DEPT	TABLE
⋮	⋮	⋮
SYS	USER_CATALOG	VIEW
PUBLIC	USER_CATALOG	SYNONYM
PUBLIC	CAT	SYNONYM
SYS	ALL_CATALOG	VIEW
PUBLIC	ALL_CATALOG	SYNONYM
⋮	⋮	⋮

# Users and Schemas in Oracle

- In Oracle, database objects (like tables) are globally identified by their owner and their name.

The owner is the user who created the object.

- Different users can create tables with the same name, these are different tables in Oracle.

I.e. every user has his/her own database schema. In Oracle, there is a 1:1-mapping between DB schemas and users (accounts). Of course, it is possible that the same person has two separate user accounts.

- If one has read access to the table DEPT owned by SCOTT, one can access it with

```
SELECT * FROM SCOTT.DEPT
```

# Synonyms in Oracle (1)

- The data dictionary is owned by the user **SYS**.

SYS is the most powerful account in Oracle.

- E.g., one can query ALL\_CATALOG with

```
SELECT *  
FROM   SYS.ALL_CATALOG  
WHERE  OWNER = 'SCOTT'
```

- However, Oracle has introduced synonyms (abbreviations, macros) to simplify this. E.g., try

```
CREATE SYNONYM DEPARTMENTS FOR SCOTT.DEPT
```

Then “SELECT \* FROM DEPARTMENTS” means actually

```
SELECT * FROM SCOTT.DEPT
```

# Synonyms in Oracle (2)

- Normal synonyms are only applicable for commands of the user who created them.
- However, Oracle also has “public synonyms”, which are available to all users (who do not have a table etc. of the same name).

Only a DBA can use “CREATE PUBLIC SYNONYM”. Public synonyms appear in the data dictionary as synonyms owned by the special user “PUBLIC”.

- CAT and ALL\_CATALOG are such public synonyms.

It is possible to create a table called “ALL\_CATALOG”. Then one must use “SYS.ALL\_CATALOG” in order to access the data dictionary “table”.

# Oracle Data Dictionary (1)

- There are three versions of the catalog table:
  - **USER\_CATALOG**: Table-like objects owned by the current user.

Columns are **TABLE\_NAME** and **TABLE\_TYPE**. The column **OWNER** (present in the other two versions) would always be the current user.

- **ALL\_CATALOG**: Table-like objects accessible by the current user.
  - **DBA\_CATALOG**: All table-like objects in the system.

Of course, **DBA\_CATALOG** is accessible only to DBAs.

- Most data dictionary tables in Oracle exist in three versions with the prefixes **USER**, **ALL**, and **DBA**.

# Oracle Data Dictionary (2)

- The “real” system tables have a rather unreadable format for performance reasons.

A system can use any data structure for the system data, as long as it offers a relational interface to these data. It does not necessarily have to be the same data structure as used for normal user tables.

- However, Oracle has defined many views to give a more user-friendly (and stable) interface.

The definitions of the data dictionary views are in:

```
$ORACLE_HOME/rdbms/admin/catalog.sql
```

- `USER_CATALOG` etc. are views owned by `SYS`.

They must be views, because otherwise they could not show every user a different result (the query uses the SQL-function “`USER`”).

# Oracle Data Dictionary (3)

- In addition, Oracle has defined public synonyms that simplify the access to these views.
- E.g., `USER_CATALOG` is a public synonym that points to `SYS.USER_CATALOG`.

The public synonyms are not defined in all Oracle installations (at least for the DBA-views). Then one must write, e.g., `SYS.USER_CATALOG`.

- Furthermore, abbreviations have been defined (as public synonyms) for the most important data dictionary tables.
- E.g., `CAT` is a synonym for `SYS.USER_CATALOG`.



# Oracle Data Dictionary (4)

- The data dictionary contains also “Dynamic Performance Views”. Their names start with **V\$**.

In contrast, the above data dictionary views are called the “Static Data Dictionary Views”.

- These “tables” give a relational interface to data structures of the server (in main memory).
- E.g., currently active sessions (users logged into Oracle) are listed in **V\$SESSION**.

Dynamic performance views use the singular form, whereas the static data dictionary views normally use the plural form (e.g., **USER\_TABLES**). This is inconsistent and confusing.

# Oracle Data Dictionary (5)

- The Oracle Data Dictionary is documented in the “Oracle Database Reference”.

Don't mix this up with the “Oracle Database SQL Reference”. Part II is about Static Data Dictionary Views, Part III about Dynamic Performance Views. For an introduction, see Chapter 7 of the Oracle Database Concepts Manual. The real system tables are not documented. However, one could analyze the view definitions.

- The data dictionary tables are read-only to ensure consistency.

E.g. INSERT cannot be used on system tables. They can only be changed with specialized commands like CREATE TABLE. As SYS, it might be possible to update the real system tables, but it is quite likely that this will destroy the entire database.

# Data Dictionary (1)

- **DICT** lists all data dictionary tables/views:

DICT	
TABLE_NAME	COMMENTS
ALL_CATALOG	All tables, views, synonyms, sequences accessible to the user
USER_CATALOG	Tables, Views, Synonyms and Sequences owned by the user
DICTIONARY	Description of data dictionary tables and views
DICT_COLUMNS	Description of columns in data dictionary tables and views
DICT	Synonym for DICTIONARY
⋮	⋮

# Data Dictionary (2)

- Columns of **DICT** are:
  - **TABLE\_NAME**: Name of the table, view, synonym.
  - **COMMENTS**: Short description.
- In Oracle 9.2.0, it has 508 rows when queried as normal user, and 1284 rows when queried as DBA.
- It is difficult to remember all data dictionary tables, but if one only remembers **DICT** and **DICT\_COLUMNS**, one has a good chance to find the right table.
- **DICT** and **DICT\_COLUMNS** contain meta-meta data.

The schema of the data dictionary. There are no (meta)<sup>3</sup>-data tables.

# Data Dictionary (3)

- E.g. this query prints all data dictionary objects containing “CAT” in their name:

```
SELECT *  
FROM   DICT  
WHERE  TABLE_NAME LIKE '%CAT%'
```

- The output in SQL\*Plus looks better if the following formatting commands are entered before the query (works only in SQL\*Plus, is not part of SQL):

```
COLUMN TABLE_NAME FORMAT A25  
COLUMN COMMENTS FORMAT A50 WORD WRAP  
SET PAGESIZE 100
```

# Data Dictionary (4)

- **DICT\_COLUMNS** contains information about the single columns of the data dictionary tables (views):

DICT_COLUMNS		
<u>TABLE_NAME</u>	<u>COLUMN_NAME</u>	COMMENTS
DICT	TABLE_NAME	Name of the object
DICT	COMMENTS	Text comment on the object
DICT_COLUMNS	TABLE_NAME	Name of the object that contains the column
DICT_COLUMNS	COLUMN_NAME	Name of the column
DICT_COLUMNS	COMMENTS	Text comment on the object
⋮	⋮	⋮

It has 13 375 entries for the DBA, 10 554 for normal users.

# Contents

- 1 Introduction
- 2 The Oracle Data Dictionary
- 3 DB Schema Information in Oracle

# Database Objects (1)

- **USER\_OBJECTS** (synonym **OBJ**) lists all database objects (tables etc. like in CAT, but also e.g. indexes, procedures, triggers) owned by the current user:

OBJ				
OBJECT_NAME	...	OBJECT_TYPE	CREATED	...
DEPT	...	TABLE	29-JAN-98	...
PK_DEPT	...	INDEX	29-JAN-98	...
EMP	...	TABLE	29-JAN-98	...
PK_EMP	...	INDEX	29-JAN-98	...
⋮	⋮	⋮	⋮	⋮



# Database Objects (2)

- The most important columns of **OBJ** are:

- **OBJECT\_NAME**: Name of the table, index, etc.
- **OBJECT\_TYPE**: E.g. TABLE, INDEX.

OBJECT\_TYPE can be: CLUSTER, FUNCTION, INDEX, LIBRARY,  
PACKAGE, PACKAGE BODY, PROCEDURE, SEQUENCE, SYNONYM, TABLE,  
TRIGGER, TYPE, UNDEFINED, VIEW.

- **CREATED**: Date/Time when object was created.
- **LAST\_DDL\_TIME**: Last ALTER TABLE, GRANT, etc.
- **TIMESTAMP**: Last change of object specification.

This changes e.g. when a column is added, but it does not change  
when constraints are added or a grant is made.

# Database Objects (3)

- Columns of **OBJ**, continued:

- GENERATED**: Was object name system generated?

E.g. when the user does not specify a constraint name for a primary key, the name of the corresponding index will be something like “SYS\_C001284”, and this column will contain a “Y”.

- STATUS**: VALID, INVALID, or N/A.

Normally, it is “VALID”. But if e.g. a view references a table that is deleted, the view is not automatically deleted, but its status becomes “INVALID”.

- TEMPORARY**: No multi-user sync., no recovery.

Each process/session can see only the data it has placed itself in the object.

# Database Objects (4)

- Of course, there are also `ALL_OBJECTS/DBA_OBJECTS` that list all accessible/all objects of the database.

These have also a column `OWNER`. All columns: `OWNER`, `OBJECT_NAME`, `SUBOBJECT_NAME`, `OBJECT_ID`, `DATA_OBJECT_ID`, `OBJECT_TYPE`, `CREATED`, `LAST_DDL_TIME`, `TIMESTAMP`, `STATUS`, `TEMPORARY`, `GENERATED`.

- E.g. when was the table “EMP” created?

```
SELECT CREATED
FROM   OBJ
WHERE  OBJECT_NAME = 'EMP'
```

- To see also the time, select the following:

```
TO_CHAR(CREATED, 'DD.MM.YYYY HH24:MI:SS')
```

# Table Columns (1)

- **USER\_TAB\_COLUMNS** (synonym **COLS**) describes the columns of tables owned by the current user:

COLS					
TABLE_NAME	COLUMN_NAME	DATA_TYPE	...	COLUMN_ID	...
DEPT	DEPTNO	NUMBER	...	1	...
DEPT	DNAME	VARCHAR2	...	2	...
DEPT	LOC	VARCHAR2	...	3	...
EMP	EMPNO	VARCHAR2	...	1	...
EMP	ENAME	VARCHAR2	...	2	...
⋮	⋮	⋮	⋮	⋮	⋮
EMP	DEPTNO	NUMBER	...	8	...
⋮	⋮	⋮	⋮	⋮	⋮

In Oracle, NUMERIC is called NUMBER, and VARCHAR2 is currently used instead of VARCHAR. Of course, Oracle understands the SQL-92 type names and internally translates them to its native types.

# Table Columns (2)

- The most important columns of COLS are:
  - **TABLE\_NAME, COLUMN\_NAME**: Identify the column.
  - **COLUMN\_ID**: Column position (1,2,...) in table.
  - **DATA\_TYPE**: E.g., CHAR, VARCHAR2, NUMBER, DATE.
  - **DATA\_PRECISION, DATA\_SCALE**: For numeric types.

DATA\_PRECISION is the total number of decimal digits, DATA\_SCALE the number of digits after the decimal point. For FLOAT, binary digits are counted in DATA\_PRECISION, and DATA\_SCALE is null.

- **CHAR\_COL\_DECL\_LENGTH**: Length of string types.
- **DATA\_LENGTH**: Maximum column length in bytes.
- **NULLABLE**: “N” if “NOT NULL”, “Y” otherwise.

# Table Columns (3)

- E.g., list all columns of the table “DEPT”:

```
SELECT COLUMN_ID, COLUMN_NAME
FROM    COLS
WHERE   TABLE_NAME = 'DEPT'
ORDER  BY COLUMN_ID
```

- In SQL\*Plus, the following command shows the columns of a table together with their types:

```
DESCRIBE <Table>
```

- As can be expected, there are also `ALL_TAB_COLUMNS` and `DBA_TAB_COLUMNS`.

# Table Columns (4)

- In total, **COLS** has 25 columns.

TABLE\_NAME, COLUMN\_NAME, DATA\_TYPE, DATA\_TYPE\_MOD,  
DATA\_TYPE\_OWNER, DATA\_LENGTH, DATA\_PRECISION, DATA\_SCALE,  
NULLABLE, COLUMN\_ID, DEFAULT\_LENGTH, DATA\_DEFAULT, NUM\_DISTINCT,  
LOW\_VALUE, HIGH\_VALUE, DENSITY, NUM\_NULLS, NUM\_BUCKETS,  
LAST\_ANALYZED, SAMPLE\_SIZE, CHARACTER\_SET\_NAME,  
CHAR\_COL\_DECT\_LENGTH, GLOBAL\_STATS, USER\_STATS, AVG\_COL\_LEN.

The sequence of columns is historically determined: Extensions at the end.

- Especially, **COLS** also contains statistical information about the columns that is used by the optimizer.

E.g. **NUM\_DISTINCT** contains the number of distinct column values. But this information is not kept current for performance reasons: Every transaction would need to lock these data. One must use e.g. the command “**ANALYZE TABLE DEPT COMPUTE STATISTICS**”, to create or update the statistical information for the columns of **DEPT** (see below).

# Constraints (1)

- **USER\_CONSTRAINTS** lists all constraints on tables that are owned by the current user.

USER_CONSTRAINTS				
OWNER	CONSTRAINT_NAME	CONSTRAINT_TYPE	TABLE_NAME	...
SCOTT	PK_DEPT	P	DEPT	...
SCOTT	SYS_C001293	C	DEPT	...
SCOTT	PK_EMP	P	EMP	...
SCOTT	FK_DEPTNO	R	EMP	...
:	:	:	:	:

- The columns in a key etc. are listed in the table **USER\_CONS\_COLUMNS**, see below.



# Constraints (2)

- Columns of **USER\_CONSTRAINTS** (slide 1/4):

- OWNER**: Owner of constraint definition.

This seems to be always the same as the owner of the table. Even if user A gives the ALTER right on a table to user B, and user B adds a constraint, still A is listed as owner. Also, even ALL\_CONSTRAINTS has not two owner columns (one for the table and one for the constraint).

- CONSTRAINT\_NAME**: Name of the constraint.

- CONSTRAINT\_TYPE**: E.g. "P" for primary key.

The complete list of type codes is: **C** for a check constraint (includes NOT NULL), **P** for primary key, **U** for unique constraint, **R** for a foreign key, **V** for "with check option" in a view declaration, **O** for "with read only" in a view declaration.

# Constraints (3)

- Columns of **USER\_CONSTRAINTS** (slide 2/4):
  - TABLE\_NAME**: Table on which constraint is defined.
  - SEARCH\_CONDITION**: Text of the CHECK-condition.

NOT NULL constraints have "A IS NOT NULL".

- R\_OWNER** and **R\_CONSTRAINT\_NAME**: Referenced key constraint (for foreign key constraints).

I.e. in order to print the referenced table of a foreign key constraint, one needs to consider two rows in USER\_CONSTRAINTS: One row (X) for the foreign key, and one (Y) for the referenced key.

Y.TABLE\_NAME is the result. Join condition: X.R\_OWNER = Y.OWNER  
AND X.R\_CONSTRAINT\_NAME = Y.CONSTRAINT\_NAME.

- DELETE\_RULE**: CASCADE or NO ACTION.

# Constraints (4)

- Columns of **USER\_CONSTRAINTS** (slide 3/4):

- STATUS**: ENABLED or DISABLED.

DISABLED means that the constraint is not checked for new or modified rows. I.e. it is still in the data dictionary, but is currently not applied. RELY (see below) together with ENABLED would also mean that the constraint is not checked.

- DEFERRABLE**: DEFERRABLE or NOT DEFERRABLE.

For a deferrable constraint, the user can choose to check it at the end of the transaction instead immediately after every statement.

- DEFERRED**: IMMEDIATE or DEFERRED.

Default setting for checking this constraint (until the user specifies something different for his transaction: SET CONSTRAINTS). For a NOT DEFERRABLE constraint, it can only be IMMEDIATE.

# Constraints (5)

- Columns of **USER\_CONSTRAINTS** (slide 4/4):

- VALIDATED**: VALIDATED or NOT VALIDATED.

VALIDATED means that all data obeys the constraint. It is possible to change the constraint from DISABLED to ENABLED without validating the existing data.

- GENERATED**: USER NAME or GENERATED NAME.
- BAD**: Possible Year-2000 problem.
- RELY** (new in 8i): System assumes that the constraint is satisfied without actually checking it.

This might be important for query optimization.

- LAST\_CHANGE**: When was it enabled/disabled?

# Constraints (6)

- **USER\_CONS\_COLUMNS**: Columns of a key or foreign key, or referenced in CHECK/NOT NULL constraints.

USER_CONS_COLUMNS				
OWNER	CONSTRAINT_NAME	TABLE_NAME	COLUMN_NAME	POSITION
BRASS	PK_STUDENTS	STUDENTS	SID	1
BRASS	PK_RESULTS	RESULTS	SID	1
BRASS	PK_RESULTS	RESULTS	CAT	2
BRASS	PK_RESULTS	RESULTS	ENO	3
BRASS	FK_RES_STUD	RESULTS	SID	1
BRASS	FK_RES_EX	RESULTS	CAT	1
BRASS	FK_RES_EX	RESULTS	ENO	2
⋮	⋮	⋮	⋮	⋮

# Constraints (7)

- Columns of **USER\_CONS\_COLUMNS**:

- OWNER, CONSTRAINT\_NAME**: Identify the constraint.
- TABLE\_NAME**: Table on which constraint is defined.

Redundant: Same as in **USER\_CONSTRAINTS**.

- COLUMN\_NAME**: Column that participates in key, foreign key, or CHECK-constraint (includes NOT NULL).
- POSITION**: Sequence number of column in key.  
1 for the first column of a composed key or foreign key, 2 for the second, and so on. The column sequence is not necessarily the same as the sequence in the table (although that should be avoided).  
**POSITION** is null for CHECK-constraints.

# Constraints (8)

- E.g. print composed primary key of table “XYZ”:

```
SELECT COL.POSITION, COL.COLUMN_NAME
FROM   USER_CONSTRAINTS CON,
       USER_CONS_COLUMNS COL
WHERE  CON.TABLE_NAME = 'XYZ'
AND    CON.CONSTRAINT_TYPE = 'P'
AND    CON.OWNER = COL.OWNER
AND    CON.CONSTRAINT_NAME = COL.CONSTRAINT_NAME
ORDER BY COL.POSITION
```

- Exercise: Print all tables that contain a foreign key that references table “XYZ”.

# Views

- **USER\_VIEWS** contains the view-defining queries:

USER_VIEWS			
VIEW_NAME	TEXT_LENGTH	TEXT	...
SALESMEN	62	SELECT ENAME, SAL+COMM AS SAL FROM EMP WHERE JOB = 'SALESMAN'	...

The column TEXT contains the view-defining query. It has data type LONG (many restrictions, e.g. it cannot be input for string concatenation "||"). In SQL\*Plus, use e.g. "SET LONG 10000" to see queries up to 10000 characters. TEXT\_LENGTH is the string length of the query.

- **COLS**: Shows columns also of views.
- **USER\_DEPENDENCIES**: Dependencies of views and procedures on tables etc. (tables used in a view).



# Synonyms

- Suppose user SCOTT creates a synonym with:

```
CREATE SYNONYM PRES FOR BRASS.PRESIDENTS
```

- **USER\_SYNONYMS** (or **SYN**) list all synonyms that were created by the current user:

USER_SYNONYMS			
SYNONYM_NAME	TABLE_OWNER	TABLE_NAME	DB_LINK
PRES	BRASS	PRESIDENTS	

- **ALL\_SYNONYMS** lists all accessible synonyms.
- **PUBLICSYN** lists all public synonyms.

# Comments (1)

- It is possible to store some documentation about tables and columns in the data dictionary:

```
COMMENT ON TABLE <Table> IS '<Text>'  
COMMENT ON COLUMN <Table>.<Column> IS '<Text>'
```

These commands are Oracle-specific.

- USER\_TAB\_COMMENTS** contains comments about own tables and views:

USER_TAB_COMMENTS		
TABLE_NAME	TABLE_TYPE	COMMENTS
EMP	TABLE	List of all employees
⋮	⋮	⋮

# Comments (2)

- **USER\_COL\_COMMENTS** contains comments about the columns of one's own tables and views:

USER_COL_COMMENTS		
TABLE_NAME	COLUMN_NAME	COMMENTS
EMP	EMPNO	Employee number (ID)
⋮	⋮	⋮

- All tables and all columns are listed.  
If no comment was stored, a null value appears in the column "COMMENTS".

Comments can be up to 4000 characters long.

# References

- Ramez Elmasri, Shamkant B. Navathe: Fundamentals of Database Systems, 3rd Ed. Chapter 17: "Database System Architectures and the System Catalog", Chapter 10: "Examples of Relational Database Management Systems: Oracle and Microsoft Access"
- Raghu Ramakrishnan, Johannes Gehrke: Database Management Systems, 2nd Ed. McGraw-Hill, 2000, ISBN 0-07-232206-3, 906 pages.
- H. Garcia-Molina, J. D. Ullman, J. Widom: Database System Implementation. Prentice Hall, ISBN 0130402648, 672 pages, ca. \$60.00
- Jason S. Couchman: Oracle8i Certified Professional: DBA Certification Exam Guide with CDROM. Osborne/ORACLE Press, ISBN 0-07-213060-1, ca. 1257 pages, ca. \$99.99.
- Jim Gray, Andreas Reuter: Transaction Processing: Concepts and Techniques. Morgan Kaufmann Publishers, 1993, ISBN 1-55860-190-2, 1070 pages, ca. \$84.95
- Oracle 8i Concepts, Release 2 (8.1.6), Oracle Corporation, 1999, Part No. A76965-01.
- Oracle 8i Administrator's Guide, Release 2 (8.1.6), Oracle Corporation, 1999, Part No. A76956-01.