Part 16: Data Dictionaries

References:

- Elmasri/Navathe: Fundamentals of Database Systems, 3rd Edition, 1999/2000.
  Chapter 17: "Database System Architectures and the System Catalog",
  Chapter 10: "Examples of Relational DBMSs: Oracle and Microsoft Access"
- Ramakrishnan/Gehrke: Database Management Systems, 2nd Edition. McGraw-Hill, 2000.
- Garcia-Molina/Ullman/Widom: Database System Implementation.
  Prentice Hall, 2000.
- Couchman: Oracle8i Certified Professional: DBA Certification Exam Guide with CDROM.
  Osborne/ORACLE Press, ISBN 0-07-213060-1, ca. 1257 pages, ca. $99.99.
- 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, 1999, Part No. A76956-01.
- Oracle8i Reference, Release 2(8.1.6), Oracle Corporation, 1999, Part No. A76961-01.
- Sunderraman: Oracle Programming, A Primer. Addison-Wesley, 1999.
- Chamberlin: A Complete Guide to DB2 Universal Database. Morgan Kaufmann, 1998.
- Michael Gertz: Oracle/SQL Tutorial, 1999.
  [http://www.db.cs.ucdavis.edu/teaching/sqltutorial/]
- Microsoft SQL Server Books Online: Accessing and Changing Data.
- Date/Darwen: A Guide to the SQL Standard, Fourth Edition, Addison-Wesley, 1997.

Objectives

After completing this chapter, you should be able to:

- explain what kind of information is typically stored in data dictionaries, and how the tables look like.
- enumerate at least three tables (or really views) from the Oracle data dictionary.
- write SQL queries that refer to the data dictionary (given the necessary table and column names).

  You need to understand that meta-data (schema information) can be represented as data in the system catalog. This is in the beginning difficult for many students.

Overview

1. General Remarks
2. Oracle Data Dictionary
3. DB2 Data Dictionary
4. SQL Server / SQL-92 Information Schema

Example: User Tables

STUDENTS

SID	FIRST	LAST	EMAIL
101	Ann	Smith	···
102	Michael	Jones	(null)
103	Richard	Turner	···
104	Maria	Brown	···

RESULTS

SID	CAT	ENO	POINTS
101	H	1	10
101	H	2	8
101	M	1	12
102	H	1	9
102	H	2	9
102	M	1	10
103	H	1	5
103	M	1	7

EXERCISES

CAT	ENO	TOPIC	MAXPT
H	1	Rel. Algeb.	10
H	2	SQL	10
M	1	SQL	14

Example: System Tables (1)

- Schema data ist often made available in system tables, e.g. there might be a table that contains a list of all tables stored in the DBMS:

SYS_TABLES			
TID	TABLE_NAME	OWNER	CREATED
1	SYS_TABLES	SYS	(null)
2	SYS_COLUMNS	SYS	(null)
3	STUDENTS	BRASS	2003-05-01
4	EXERCISES	BRASS	2003-05-01
5	RESULTS	BRASS	2003-05-01

The names and structure of system tables depend very much on the DBMS, this is only an example.

Example: System Tables (2)

SYS_COLUMNS						
TID	SEQ	COLUMN_NAME	TYPE	LENGTH	PREC	NULL
1	1	TID	NUMERIC	5	0	N
1	2	TABLE_NAME	VARCHAR	128	(null)	N
1	3	OWNER	VARCHAR	128	(null)	N
1	4	CREATED	DATE	(null)	(null)	Y
⋮	⋮	⋮	⋮	⋮	⋮	⋮
3	1	SID	NUMERIC	3	0	N
3	2	FIRST	VARCHAR	20	(null)	N
3	3	LAST	VARCHAR	20	(null)	N
3	4	EMAIL	VARCHAR	80	(null)	Y
⋮	⋮	⋮	⋮	⋮	⋮	⋮

Example: System Tables (3)

- It might be at first difficult to grasp the idea that names of schema objects are now stored as data.
- Such data is called "meta-data" (data about data).

  One might fear that there is an infinite recursion: "meta-meta-data" and so on. This is solved by storing the system table names in the system tables itself. Oracle actually has in addition a list of system tables (called `DICT`), but no higher level (would not be interesting).

- E.g. the following query lists all tables owned by the user "Brass":

```
SELECT TABLE_NAME
FROM   SYS_TABLES
WHERE  OWNER = 'BRASS'
```


Data Dictionaries (1)

- The collection of system tables is called the "data dictionary" or the "system catalog".

  Maybe the name "Data Dictionary" was chosen because it gives meaning to the stored data by defining tables and columns.

- A good DBMS makes all its information available in the data dictionary.

  Thus, to understand the information in the data dictionary is to understand the system. All parameters are reflected in the data dictionary.

- The data dictionary is a really important tool for the database administrator (DBA).

Data Dictionaries (2)

Information in the Data Dictionary:

- Tables, views, etc. (schema objects).
- Comments about tables and columns.
- Database Users, Access Rights.
- A log of executed commands (if auditing is done).
- Indexes, Physical Storage Parameters.
- File space usage

  E.g. which disks blocks are allocated for a given table, how much space is still free?
- Statistical Information, Performance Data.

Data Dictionaries (3)

- With the data dictionary, 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 for listing tables and columns.
- E.g., in Oracle SQL*Plus, "`describe` ⟨`Table`⟩" lists all columns of a given table.

  Internally, this is actually executed as a query to the data dictionary.
- However, this command would not help if a table with a given column is searched.

Data Dictionaries (4)

- Many of the data dictionary tables are only available to users with DBA rights (administrators).
- Also, 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`.
- The data dictionary is 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).

Data Dictionaries (5)

- A DBMS can use any data structure for the system data, and offer a relational interface to these data.

  It does not necessarily have to be the same data structure as used for normal user tables.
- However, at least some systems actually store the system data in normal tables.

  Then there is a kind of bootstrapping problem: How can one access these tables without knowing their contents? E.g. the system catalog also contains the addresses of the disk blocks used for each table. This problem can be solved by making sure that the most important system tables are stored at fixed addresses when a new database is created. These addresses and some other important information about the system tables are then built into the DBMS software.

Data Dictionaries (6)

- In Oracle, the "real" system tables have a rather unreadable format for performance reasons.
- Oracle has defined many views to give a more user-friendly interface. The definitions are contained in:

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

- There are also graphical tools for browsing the data dictionary (e.g. Oracle Enterprise Manager).

  Of course, the OEM also permits to change some parameters and perform system administration tasks. Try also the free tool ODDIS! [http://www-db.informatik.uni-hannover.de/software/oddis/]

Overview

1. General Remarks
2. Oracle Data Dictionary
3. DB2 Data Dictionary
4. SQL Server / SQL-92 Information Schema

Tables, Views, etc. (1)

- CAT (short for USER_CATALOG) lists all table-like objects (tables, views, sequences, synonyms) owned by the current user.
- E.g., suppose that the current Oracle user "BRASS" has created the three example tables on Slide 16-4. Then CAT looks as follows:

CAT	
TABLE_NAME	TABLE_TYPE
STUDENTS	TABLE
EXERCISES	TABLE
RESULTS	TABLE

Tables, Views, etc. (2)

- The following query lists all tables owned by the current user:

```
SELECT TABLE_NAME
FROM   CAT
WHERE  TABLE_TYPE = 'TABLE'
```

- Obviously, CAT is a view (not a stored table).

  Its contents depends on the current user. This can be done by using the 0-adic function "USER" in the defining query. Actually, CAT is a synonym. Synonyms are alternative names (abbreviations) for tables etc. They are an Oracle extension to the SQL standard.

- CAT itself is not listed because it is not owned by the current user BRASS (it is owned by SYS).

Tables, Views, etc. (3)

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

  Whereas in SQL, keywords and table/column/user names are case-insensitive, they appear as string data in the data dictionary. Depending on the DBMS and possibly certain parameters, string comparisons are case-sensitive (e.g. in Oracle). Normally, all identifiers are automatically translated to uppercase when they are stored in the data dictionary (more precisely, when the `CREATE TABLE` or other SQL statement is parsed).

- E.g. the result of the following query will be empty:

```
SELECT TABLE_TYPE     Wrong!
FROM   CAT
WHERE  TABLE_NAME = 'students'
```


Tables, Views, etc. (4)

- `ALL_CATALOG` lists all table-like objects that are accessible by the current user:

ALL_CATALOG		
<u>OWNER</u>	<u>TABLE_NAME</u>	TABLE_TYPE
BRASS	STUDENTS	TABLE
BRASS	EXERCISES	TABLE
⋮	⋮	⋮
SYS	USER_CATALOG	VIEW
PUBLIC	USER_CATALOG	SYNONYM
PUBLIC	CAT	SYNONYM
SYS	ALL_CATALOG	VIEW
PUBLIC	ALL_CATALOG	SYNONYM
⋮	⋮	⋮

Tables, Views, etc. (5)

- In Oracle, database objects (tables, views, etc.) are uniquely identified by owner and object name.

  The owner is the user who created the table/view/etc. Every user has exactly one schema, and every schema belongs to exactly one user. Two distinct users can have tables with the same name.

- If one has the "`SELECT`" access right for a table owned by another user, one can refer to this table by prefixing the user name and a ".", e.g.

```
SELECT *
FROM   BRASS.EXERCISES
```


Tables, Views, etc. (6)

- `ALL_CATALOG` lists all tables etc. for which the current user has any access right (at least one of `SELECT`, `INSERT`, `DELETE`, or `UPDATE`) — possibly via a role.

  If a user has no right to access a table, he/she should not even know that the table exists. Therefore, Oracle prints the error message "Table or view does not exist" even if the table actually exists, but the user has no access rights for it.

- In our current database, the query for all accessible tables, views, etc. lists 1365 database objects:

```
SELECT * FROM ALL_CATALOG
```

- Most of these are objects from the data dictionary.

Tables, Views, etc. (7)

- ALL_CATALOG itself appears twice in ALL_CATALOG:
  - ◇ As a view owned by SYS.

    SYS is a user created by Oracle that owns the data dictionary.
  - ◇ As a synonym owned by PUBLIC (public synonym).

    Public synonyms can be applied by all users, even if they are not owned by them. Otherwise, if one refers to objects defined by other users, one must include the username as shown above.
- It is possible to access the view directly:

```
SELECT * FROM SYS.ALL_CATALOG
```

- However, the synonym ALL_CATALOG was declared as an abbreviation for SYS.ALL_CATALOG.

Tables, Views, etc. (8)

Naming Conventions for Data Dictionary Tables/Views:

- USER_*: Objects for which the current user is owner.

  There are shorter synonyms declared for the most important of these tables, e.g. CAT for USER_CATALOG.
- ALL_*: Objects accessible by the current user.
- DBA_*: All objects of the DB.

  This view can be accessed by the DBA only. Depending on the installation, it might be necessary to write, e.g. "SYS.DBA_CATALOG".
- V$*: Dynamic performance views.

  Again, only for the DBA. These "tables" give a relational interface to data structures of the server. They are not actually stored tables.

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 8.1.6, it has 305 rows when queried as normal user, and 764 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.

  Seasoned Oracle DBAs probably know more than 50 tables.
  See also: Oracle8i Reference, Ch. 2: Static Data Dictionary Views.

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, 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		
TABLE_NAME	COLUMN_NAME	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 8285 entries for the DBA, 6681 for normal users.

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	···
STUDENTS	···	TABLE	29-JAN-98	···
PK_STUD	···	INDEX	29-JAN-98	···
EXERCISES	···	TABLE	29-JAN-98	···
PK_EX	···	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 "STUDENTS" created?

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

- 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	···	COLUM_ID	···
STUDENTS	SID	NUMBER	···	1	···
STUDENTS	FIRST	VARCHAR2	···	2	···
STUDENTS	LAST	VARCHAR2	···	3	···
STUDENTS	EMAIL	VARCHAR2	···	4	···
EXERCISES	CAT	CHAR	···	1	···
EXERCISES	ENO	NUMBER	···	2	···
⋮	⋮	⋮	⋮	⋮	⋮

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. Sequence 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 STUDENTS COMPUTE STATISTICS", to create or update the statistical information for the columns of STUDENTS.

Storage Information (1)

- USER_TABLES (synonym TABS) contains information about base tables (i.e. not views):

TABS					
TABLE_NAME	TABLESPACE_NAME	···	NUM_ROWS	BLOCKS	···
STUDENTS	USERS	···	(null)	(null)	···
EXERCISES	USERS	···	(null)	(null)	···
RESULTS	USERS	···	(null)	(null)	···
⋮	⋮	⋮	⋮	⋮	⋮

- TABS is only interesting for storage information: Table names are already contained in CAT.

  CAT also containes view etc., but TABLE_TYPE = 'TABLE' selects only the base tables.

Storage Information (2)

- TABS contains the physical storage parameters for every base table.

  These parameters are specified in the CREATE TABLE command and are Oracle-specific. If one has not defined them, the default values are used, which are then shown in TABS.

- Some columns describing storage parameters are:
  - ◇ TABLESPACE_NAME: Place where the table is stored.

    A tablespace is a "logical disk" consisting of one or more database files. See also: USER_TS_QUOTAS(TABLESPACE_NAME, ...).
  - ◇ INITIAL_EXTENT: Size of first chunk of storage allocated for the table.

Storage Information (3)

- In addition, TABS contains information about the table size to be used by the optimizer.

  As for COLS, e.g. "ANALYZE TABLE STUDENTS COMPUTE STATISTICS" must be used to compute this information and store it in the data dictionary.

- Some columns containing statistical information for the optimizer:
  - ◇ NUM_ROWS: Number of rows in the table.
  - ◇ BLOCKS: Number of used data blocks in the table.

    In our DB, the parameter db_block_size is set to 8 KB.
  - ◇ AVG_ROW_LEN: Average length of a row in bytes.

Quotas (1)

- USER_TS_QUOTAS: How many bytes/blocks on which tablespace are allocated for tables of the current user, and what is the allowable maximum (quota)?

USER_TS_QUOTAS

TABLESPACE_NAME	BYTES	MAX_BYTES	BLOCKS	MAX_BLOCKS
TEMP	0	-1	0	-1
USERS	245760	5242880	30	2560

- This lists all tablespaces to which the current user has access, not all tablespaces that exist in the DB.
- Storage size per DB object: See USER_SEGMENTS.

Quotas (2)

- Columns of USER_TS_QUOTAS:
  - ◇ TABLESPACE_NAME: Physical storage container.
  - ◇ BYTES/BLOCKS: Amount of storage in this tablespace charged to the current user.

    The allocation is always in units of blocks, therefore one of the two is redundant. Oracle has a parameter for the blocksize, it is currently often 8 KByte. The blocks might not yet be full: E.g. when the table is created, a certain number of blocks is allocated for that table, although the table is still empty.
  - ◇ MAX_BYTES/MAX_BLOCKS: Quota for the tablespace.

    If this amount of storage is used up, further insertions will fail (after all the allocated blocks are really full). -1 means that there is no limit (i.e. the complete tablespace may be filled by this user).

Quotas (3)

- All space used by tables and indexes owned by a user is charged to that user, even if other users inserted the columns.

  Of course, that is only possible if they have the necessary access rights. The above rule makes sense since storage is anyway allocated in units of blocks.

- Related tables available to the DBA:
  - ◇ DBA_TS_QUOTAS: Storage usage by all users.
  - ◇ DBA_TABLESPACES: List of tablespaces.
  - ◇ DBA_DATA_FILES: Data files for each tablespace.
  - ◇ DBA_FREE_SPACE: Currently free pieces of storage.

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	···
BRASS	PK_STUD	P	STUDENTS	···
BRASS	SYS_C001293	C	STUDENTS	···
BRASS	PK_RES	P	RESULTS	···
BRASS	RES_REF_STUD	R	RESULTS	···
BRASS	RES_REF_EX	R	RESULTS	···
⋮	⋮	⋮	⋮	⋮

- The columns in a key etc. are listed in the table `USER_CONS_COLUMNS`, see below.

Constraints (2)

- Most important columns of `USER_CONSTRAINTS`:
  - ◇ `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)

- Important columns of `USER_CONSTRAINTS`, continued:
  - ◇ `TABLE_NAME`: Table on which constraint is defined.
  - ◇ `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`.
  - ◇ `SEARCH_CONDITION`: Text of the `CHECK`-condition.

    `NOT NULL` constraints have “A `IS NOT NULL`”.

Constraints (4)

- In total, `USER_CONSTRAINTS` has 16 columns.

  `OWNER`, `CONSTRAINT_NAME`, `CONSTRAINT_TYPE`, `TABLE_NAME`, `SEARCH_CONDITION`, `R_OWNER`, `R_CONSTRAINT_NAME`, `DELETE_RULE`, `STATUS`, `DEFERRABLE`, `DEFERRED`, `VALIDATED`, `GENERATED`, `BAD`, `RELY`, `LAST_CHANGE`.

- Print all `CHECK` constraints on the table “`EXERCISES`”:

```
SELECT SEARCH_CONDITION
FROM   USER_CONSTRAINTS
WHERE  TABLE_NAME = 'EXERCISES'
AND    CONSTRAINT_TYPE = 'C'
```

Constraints (5)

- 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 (6)

- 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 (7)

- E.g. print the primary key of "RESULTS":

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

- Exercise: Print referencing table and column and referenced table for all foreign key constraints.

  Assume that they consist only of one attribute.

Views (1)

- Suppose the following view is declared:

```
CREATE VIEW MIDTERM(STUDENT, EXERCISE, POINTS)
         AS SELECT SID, ENO, POINTS
            FROM   RESULTS
            WHERE  CAT = 'M'
```

- USER_VIEWS contains the view-defining queries:

USER_VIEWS			
VIEW_NAME	TEXT_LENGTH	TEXT	...
MIDTERM	56	SELECT SID, ENO, POINTS FROM RESULTS WHERE CAT = 'M'	... ...

Views (2)

- Selected columns of USER_VIEWS:
  - ◇ VIEW_NAME: Name of the view.
  - ◇ TEXT_LENGTH: String length of the query.
  - ◇ TEXT: Text of the view-defining query.

    This column has data type LONG (This implies many restrictions, e.g. it cannot be input for the string concatenation operator "||"). In SQL*Plus, use e.g. "SET LONG 10000" to see queries up to 10000 characters.
- In total, USER_VIEWS has 9 columns.

  VIEW_NAME, TEXT_LENGTH, TEXT, TYPE_TEXT LENGTH, TYPE_TEXT, OID_TEXT_LENGTH, OID_TEXT, VIEW_TYPE_OWNER, VIEW_TYPE.

Views (3)

- View names can also be looked up in CAT or OBJ:

```
SELECT TABLE_NAME
FROM   CAT
WHERE  TABLE_TYPE = 'VIEW'
```

- View columns are represented in COLS:

COLS					
TABLE_NAME	COLUMN_NAME	DATA_TYPE	···	COLUM_ID	···
MIDTERM	STUDENT	NUMBER	···	1	···
MIDTERM	EXERCISE	NUMBER	···	2	···
MIDTERM	POINTS	NUMBER	···	3	···
⋮	⋮	⋮	⋮	⋮	⋮

Views (4)

- USER_DEPENDENCIES: Dependencies of views and procedures on tables etc.:

USER_DEPENDENCIES				
NAME	TYPE	REFERENCED_OWNER	REFERENCED_NAME	···
MIDTERM	VIEW	BRASS	RESULTS	···

- Most important columns:
  - ◇ NAME, TYPE: Dependent object (e.g. view).
  - ◇ REFERENCED_OWNER, REFERENCED_NAME, REFERENCED_TYPE: Object that the view etc. uses.

Synonyms (1)

- Synonyms are alternative names (abbreviations) for tables, views, etc. (Oracle-specific SQL extension).
- Synonyms are e.g. used to avoid the "OWNER.TABLE" notation.

  After "CREATE SYNONYM DEPT FOR SCOTT.DEPT" one can write "DEPT", as if the table would be contained in one's own schema, although it is contained in the schema of the user "SCOTT". In the same way, one can avoid "database links" for tables that are stored in other databases.
- Public synonyms are available to all DB users.

  However, it is still possible to define a table "T", even if "T" is a public synonym. Then "T" will mean the table and not the public synonym.

Synonyms (2)

- USER_SYNONYMS (or SYN) list all synonyms that were created by the curent user:

USER_SYNONYMS			
SYNONYM_NAME	TABLE_OWNER	TABLE_NAME	DB_LINK
STUD	BRASS	STUDENTS	
DEPT	SCOTT	DEPT	

- 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
STUDENTS	TABLE	List of all Students
⋮	⋮	⋮

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
STUDENTS	SID	Student 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.

Users (1)

- ALL_USERS: List of all users, accessible by all users:
  - ◇ USERNAME: Name of the Oracle account.
  - ◇ USER_ID: Internal number of the account.
  - ◇ CREATED: Date/time when account was created.

ALL_USERS		
USERNAME	USER_ID	CREATED
SYS	0	29-JAN-98
SYSTEM	5	29-JAN-98
SCOTT	20	29-JAN-98
BRASS	24	13-MAY-01
⋮	⋮	⋮

Users (2)

- DBA_USERS: Full information about all users.
  Only the DBA can look at this table.

  It has the following columns: USERNAME, USER_ID, PASSWORD (stored in encrypted form), DEFAULT_TABLESPACE, TEMPORARY_TABLESPACE, CREATED, PROFILE, ACCOUNT_STATUS (indicates whether account is locked, expired, or unlocked), LOCK_DATE, EXPIRY_DATE, INITIAL_RSRC_CONSUMER_GROUP, EXTERNAL_NAME.

- USER_USERS: Single row with information about the current user.

  It has the following columns: USERNAME, USER_ID, ACCOUNT_STATUS, LOCK_DATE, EXPIRY_DATE, DEFAULT_TABLESPACE, CREATED, EXTERNAL_NAME.

Access Rights (1)

- USER_TAB_PRIVS: Grants on objects for which the current user is owner, grantor, or grantee.

USER_TAB_PRIVS					
GRANTEE	OWNER	TABLE_NAME	GRANTOR	PRIVILEGE	GRANTABLE
PUBLIC	BRASS	EXERCISES	BRASS	SELECT	N
MICHEL	BRASS	STUDENTS	BRASS	SELECT	N
MICHEL	BRASS	RESULTS	BRASS	SELECT	N
MICHEL	BRASS	RESULTS	BRASS	INSERT	N
MICHEL	BRASS	RESULTS	BRASS	UPDATE	N

- I.e. all users have read access to the table exercises. The GSA "MICHEL" has read access to STUDENTS, and read, insert, update rights for RESULTS.

Access Rights (2)

- Columns of USER_TAB_PRIVS:
  - ◇ OWNER and TABLE_NAME identify the table.
  - ◇ GRANTEE: The user who got the privilege.
  - ◇ GRANTOR: The user who gave the privilege.

    Because of the grant option, not only the owner can be grantor.
  - ◇ PRIVILEGE: The right, e.g. 'SELECT'.
  - ◇ GRANTABLE: 'YES' if right includes grant option.
- In this way, the SQL GRANT commands are stored in the data dictionary.

Access Rights (3)

- USER_TAB_PRIVS_MADE is the subset of USER_TAB_PRIVS with OWNER=USER.
- USER_TAB_PRIVS_RECD is the subset of USER_TAB_PRIVS with GRANTEE=USER.
- The user might also have access to database objects because of grants to PUBLIC, which are not listed in these tables.

  Unless, of course, they are made by the current user or refer to tables of the current user. Otherwise, the name of the current user is neither OWNER, nor GRANTOR, nor GRANTEE, therefore the grant is not shown.

Access Rights (4)

- The INSERT and UPDATE right can be given selectively for certain columns.

  An insert right for only part of the columns means that the other columns cannot be explicitly specified, and thus get their declared default value (or null).

- USER_COL_PRIVS: Grants that refer to single columns.

  USER_COL_PRIVS looks like USER_TAB_PRIVS, but has the additional column COLUMN_NAME.

- Grants for whole tables are not repeated here.
- USER_COL_PRIVS_MADE, USER_COL_PRIVS_RECD: Subsets with current user as owner/grantee (as above).

System Privileges (1)

- Not every database user needs to create new tables.
- But object privileges as contained in the SQL standard cannot exclude this.
- Therefore, Oracle introduced "system privileges" that specify the commands a user may execute.

  Solutions in other systems are similar.

- Another important application of system privileges is to specify exactly, what a certain DBA may do.

  Large systems often have several DBAs, and not every DBA may simply do "everything".

System Privileges (2)

- SYSTEM_PRIVILEGE_MAP: List of all system privileges.

SYSTEM_PRIVILEGE_MAP	
PRIVILEGE	NAME
⋮	⋮
-5	CREATE SESSION
⋮	⋮
-40	CREATE TABLE
⋮	⋮
-47	SELECT ANY TABLE
⋮	⋮

System Privileges (3)

- USER_SYS_PRIVS: System privileges granted to the current user or to PUBLIC.

  Columns are: USERNAME (always the name of the current user, not very useful), PRIVILEGE (name of the system privilege, no join with SYSTEM_PRIVILEGE_MAP necessary), ADMIN_OPTION (similar to grant option for object privileges).

- DBA_SYS_PRIVS: System privileges for each user.

  For DBA only. It has the columns GRANTEE, PRIVILEGE, ADMIN_OPTION.

- Only directly granted privileges are listed.

  Additional system privileges might have been granted via roles (see below). Therefore, USER_SYS_PRIVS is often empty, although the user actually has many system privileges.

Roles (1)

- Often many users with the same access rights have to be managed (user groups).
- In Oracle, this is done via roles.
- Roles are sets of privileges (object privileges and system privileges) that can be granted as a whole to users (or to other roles).

  Roles are Oracle specific. In other systems, user groups were introduced for the same purpose.
- If role A is granted to role B, B includes all rights of A. Thus, B is more powerful than A.

Roles (2)

- `DBA_ROLES`: List of all roles defined in the system.

  It has the columns `ROLE`, `PASSWORD_REQUIRED`. Only the DBA can create roles, and only the DBA can see the list of all roles.
- `USER_ROLE_PRIVS`: Roles granted to the current user.

  Roles granted to `PUBLIC` are also listed: All users have the rights included in such roles. Columns are: `USERNAME`, `GRANTED_ROLE`, `ADMIN_OPTION`, `DEFAULT_ROLE`, `OS_GRANTED`.
- `DBA_ROLE_PRIVS`: Which roles are granted to which user? Also role-to-role grants are shown.

  Columns: `GRANTEE`, `GRANTED_ROLE`, `ADMIN_OPTION`, `DEFAULT_ROLE`. `GRANTEE` can be a user or another role.

Roles (3)

- The following tables/views list the access rights included in roles accessible to the current user:
  - ◇ `ROLE_ROLE_PRIVS`: Roles implied by a role.

    Columns are: `ROLE`, `GRANTED_ROLE`, `ADMIN_OPTION`.
    All rights in `GRANTED_ROLE` are included in `ROLE`.
  - ◇ `ROLE_SYS_PRIVS`: System privileges in a role.

    Columns are: `ROLE`, `PRIVILEGE`, `ADMIN_OPTION`.
  - ◇ `ROLE_TAB_PRIVS`: Table privileges granted to roles.

    Columns are: `ROLE`, `OWNER`, `TABLE_NAME`, `COLUMN_NAME` (null if right for entire table), `PRIVILEGE`, `GRANTABLE`.

Overview

1. General Remarks
2. Oracle Data Dictionary
3. DB2 Data Dictionary
4. SQL Server / SQL-92 Information Schema

General Remarks

- An older/internal version of the data dictionary is stored in the tables of the schema `SYSIBM`.
- Views for the user are made available in the schema `SYSCAT`.

  It consists of 38 views (i.e. much smaller than Oracle).
- Views containing statistical information about table sizes etc. in the schema `SYSSTAT`.

  It consists of 5 views. The statistical information is updated by running the `RUNSTATS` utility. However, in order to influence the optimizer in specific ways, the views in `SYSSTAT` are actually updatable.

Schemas

- Information about schemas is made available in the view `SYSCAT.SCHEMATA`. with the following columns:
  - ◇ `SCHEMANAME`: Name of the schema.
  - ◇ `OWNER`: Name of the user who owns the schema.
  - ◇ `DEFINER`: User who created the schema.
  - ◇ `CREATE_TIME`: Date/Time when the schema was created.
  - ◇ `REMARKS`: Text stored with the `COMMENT ON SCHEMA` command.
- The underlying base table is `SYSIBM.SYSSCHEMATA`.

Tables

- `SYSCAT.TABLES`: Information about tables, views, etc.
- This view has e.g. the following columns:
  - ◇ `TABSCHEMA`: Schema in which the table is defined.
  - ◇ `TABNAME`: Name of the table.
  - ◇ `DEFINER`: User who created the table.
  - ◇ `TYPE`: `'T'` for tables, `'V'` for views, `'A'` for aliases.
  - ◇ `CREATE_TIME`: Date/time when table was created.
  - ◇ `COLCOUNT`: Number of columns.
  - ◇ `REMARKS`: Explanation from `COMMENT` command.

Columns

- `SYSCAT.COLUMNS`: Information about columns of tables and views. Selected colums are:
  - ◇ `TABSCHEMA/TABNAME`: Identifies the table.
  - ◇ `COLNAME`: Name of the column.
  - ◇ `COLNO`: Position of the column (starts with 0).
  - ◇ `TYPESCHEMA/TYPENAME`: Name of the data type.
  - ◇ `LENGTH`: Maximum length of the column.
  - ◇ `SCALE`: Number of digits after decimal point.
  - ◇ `DEFAULT`: Default value for this column.
  - ◇ `NULLS`: `'Y'` if column allows null values.
  - ◇ `REMARKS`: Text of `COMMENT ON COLUMN` command.

Access Rights

- SYSCAT.TABAUTH. contains information about granted privileges for tables and views. It has the columns:
  - ◇ GRANTOR: User who granted the privilege.
  - ◇ GRANTEE: User who received the privilege.
  - ◇ GRANTEETYPE: 'U' if GRANTEE is user, 'G' if group.
  - ◇ TABSCHEMA/TABNAME: Table to which right applies.
  - ◇ CONTROLAUTH, ALTERAUTH, DELETEAUTH, INDEXAUTH, INSERTAUTH, SELECTAUTH, REFAUTH, UPDATEAUTH: 'Y': privilege was granted without grant option, 'G': with grant option, 'N': not granted.

Overview

1. General Remarks
2. Oracle Data Dictionary
3. DB2 Data Dictionary
4. SQL Server / SQL-92 Information Schema

General Remarks (1)

- SQL Server has a "native" data dictionary and implements part of the SQL-92 "information schema" standard.
- The native data dictionary consists of two parts:
  - ◇ The "System Catalog" which is stored in the database "master" and contains settings for the server.
  - ◇ Each database managed by the server contains a "Database Catalog" with information for that database only.

General Remarks (2)

- The names of these system tables/views start with "sys". They are owned by "dbo", so no prefix is needed (except possibly for the database).
- The views corresponding to the SQL-92 standard are owned by the user "INFORMATION_SCHEMA".

  SQL Server has 17 such tables, the SQL-92 standard mentions 24: INFORMATION_SCHEMA_CATALOG_NAME, SCHEMATA, DOMAINS, TABLES, VIEWS, COLUMNS, TABLE_PRIVILEGES, COLUMN_PRIVILEGES, USAGE_PRIVILEGES, DOMAIN_CONSTRAINTS, TABLE_CONSTRAINTS, REFERENTIAL_CONSTRAINTS, CHECK_CONSTRAINTS, KEY_COLUMN_USAGE, ASSERTIONS, CHARACTER_SETS, COLLATIONS, TRANSLATIONS, VIEW_TABLE_USAGE, VIEW_COLUMN_USAGE, CONSTRAINT_TABLE_USAGE, CONSTRAINT_COLUMN_USAGE, COLUMN_DOMAIN_USAGE, SQL_LANGUAGES.

Tables

- Information about tables and views in the current database for which the current user has permissions are available in `INFORMATION_SCHEMA.TABLES`.
- It has the following columns:
  - ◇ `TABLE_CATALOG`: Name of the database.
  - ◇ `TABLE_SCHEMA`: Owner of the table.
  - ◇ `TABLE_NAME`: Name of the table.
  - ◇ `TABLE_TYPE`: `'VIEW'` or `'BASE TABLE'`.
- This view is based on the table `SYSOBJECTS` from the database catalog.

Columns (1)

- `INFORMATION_SCHEMA.COLUMNS`: columns in tables and views accessible by the current user in the current database.
- It has e.g. the following columns (23 in total):
  - ◇ `TABLE_CATALOG/TABLE_SCHEMA/TABLE_NAME`: Database, owner and name of the table.
  - ◇ `COLUMN_NAME`: Name of the column.
  - ◇ `ORDINAL_POSITION`: Column position.

    E.g. 1 for first/leftmost column.

Columns (2)

- Columns of `INFORMATION_SCHEMA.COLUMNS`, continued:
  - ◇ `COLUMN_DEFAULT`: Default value.
  - ◇ `IS_NULLABLE`: `'YES'` if null values are allowed.

    Otherwise `'No'` is printed (the manual says `'NO'`).
  - ◇ `DATA_TYPE`: Data type name.
  - ◇ `CHARACTER_MAXIMUM_LENGTH`: Maximum number of characters (for string types).
  - ◇ `CHARACTER_OCTET_LENGTH`: Maximum string length in bytes.

Columns (3)

- Columns of `INFORMATION_SCHEMA.COLUMNS`, continued:
  - ◇ `NUMERIC_PRECISION`: Maximal number of digits.
  - ◇ `NUMERIC_PRECISION_RADIX`: 10 if decimal digits.
  - ◇ `NUMERIC_SCALE`: Number of digits after decimal point.
- This view is based on the tables `SYSCOLUMNS` and `SYSTYPES` from the DB catalog (plus other tables).

Table Privileges (1)

- `INFORMATION_SCHEMA.TABLE_PRIVILEGES` describes access rights granted to or by the current user in the current database. It has the following columns:
  - ◇ `GRANTOR`: User who granted the privilege.
  - ◇ `GRANTEE`: User who received the privilege.
  - ◇ `TABLE_CATALOG/TABLE_SCHEMA/TABLE_NAME`: Table to which the access right applies.
  - ◇ `PRIVILEGE_TYPE`: E.g. `'SELECT'`, `'INSERT'`.
  - ◇ `IS_GRANTABLE`: `'YES'` if with grant option, `'NO'` otherwise.

Table Privileges (2)

- `INFORMATION_SCHEMA.TABLE_PRIVILEGES` is based on the following database catalog tables: `SYSPROTECTS`, `SYSOBJECTS`, `SYSUSERS`.