11

Managing Objects with Data Dictionary Views

ORACLE

Objectives

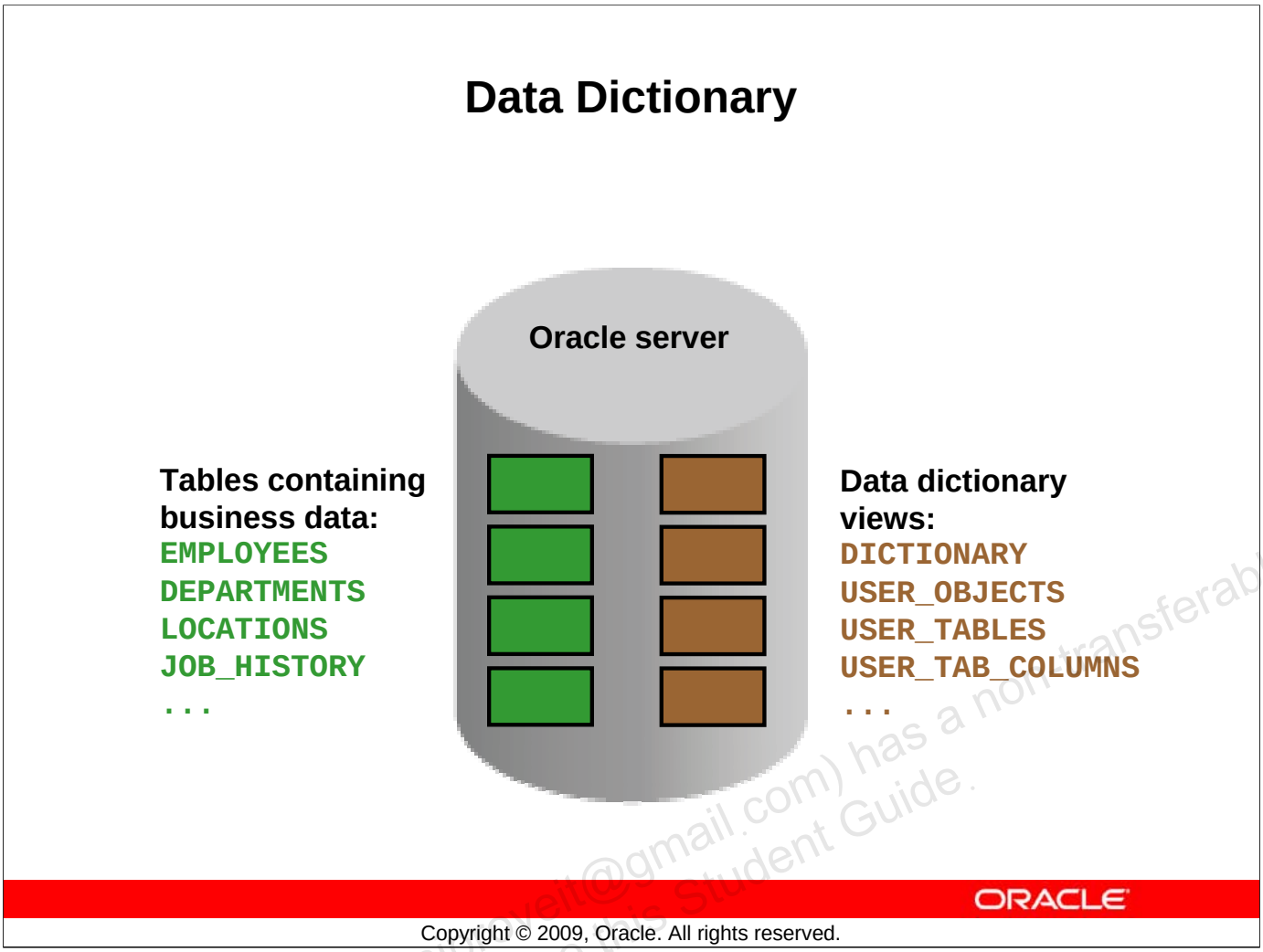
After completing this lesson, you should be able to do the following:

- Use the data dictionary views to research data on your objects
- Query various data dictionary views

ORACLE

Objectives

In this lesson, you are introduced to the data dictionary views. You learn that the dictionary views can be used to retrieve metadata and create reports about your schema objects.



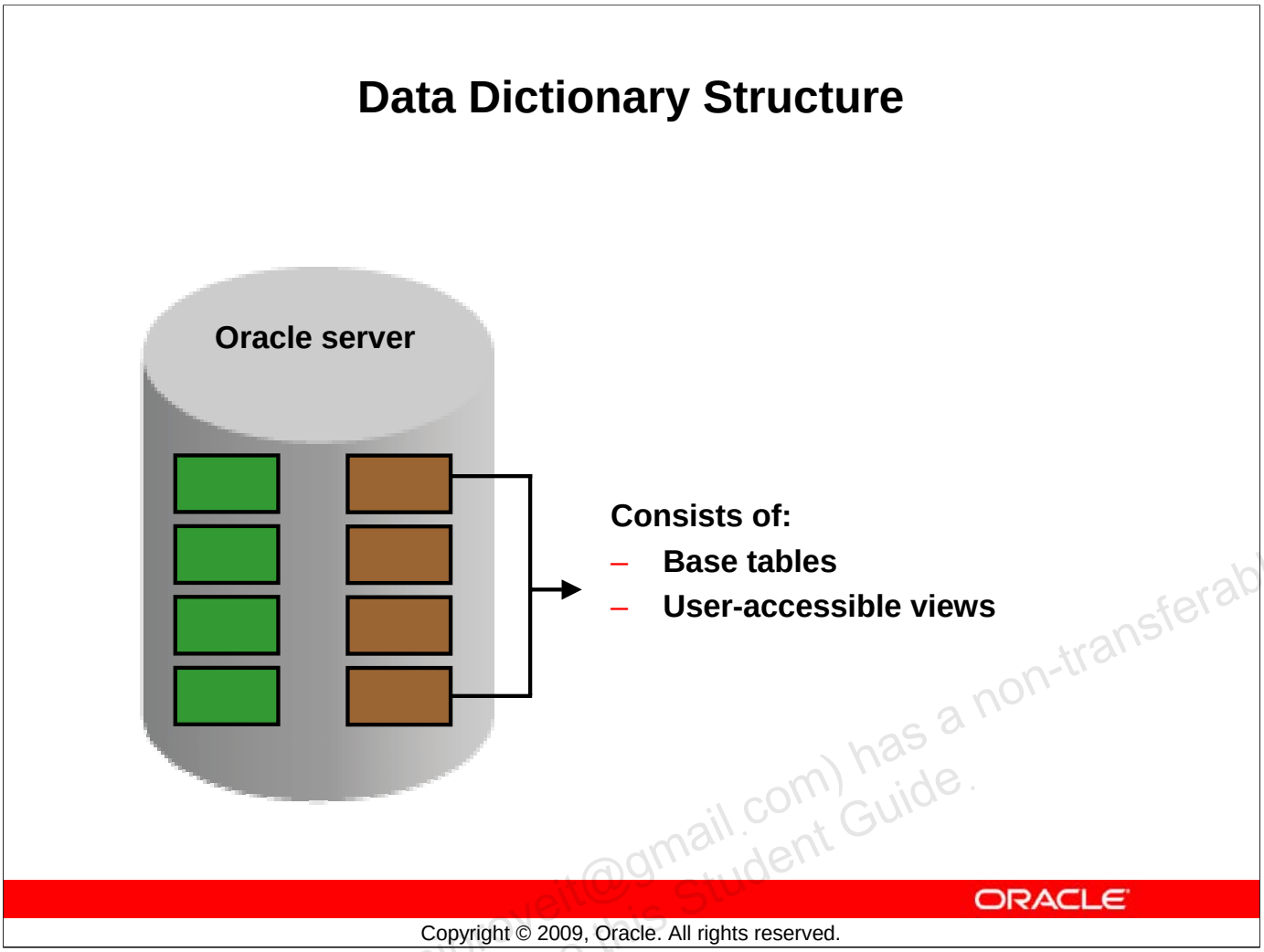
Data Dictionary

User tables are tables created by the user and contain business data, such as `EMPLOYEES`. There is another collection of tables and views in the Oracle Database known as the data dictionary. This collection is created and maintained by the Oracle server and contains information about the database. The *data dictionary* is structured in tables and views, just like other database data. Not only is the data dictionary central to every Oracle Database, but it is also an important tool for all users, from end users to application designers and database administrators.

You use SQL statements to access the data dictionary. Because the data dictionary is read-only, you can issue only queries against its tables and views.

You can query the dictionary views that are based on the dictionary tables to find information such as:

- Definitions of all schema objects in the database (tables, views, indexes, synonyms, sequences, procedures, functions, packages, triggers, and so on)
- Default values for columns
- Integrity constraint information
- Names of Oracle users
- Privileges and roles that each user has been granted
- Other general database information



Data Dictionary Structure

Underlying base tables store information about the associated database. Only the Oracle server should write to and read these tables. You rarely access them directly.

There are several views that summarize and display the information stored in the base tables of the data dictionary. These views decode the base table data into useful information (such as user or table names) using joins and `WHERE` clauses to simplify the information. Most users are given access to the views rather than the base tables.

The Oracle user `SYS` owns all base tables and user-accessible views of the data dictionary. No Oracle user should *ever* alter (`UPDATE`, `DELETE`, or `INSERT`) any rows or schema objects contained in the `SYS` schema, because such activity can compromise data integrity.

Data Dictionary Structure

View naming convention:

View Prefix	Purpose
USER	User's view (what is in your schema; what you own)
ALL	Expanded user's view (what you can access)
DBA	Database administrator's view (what is in everyone's schemas)
V$	Performance-related data

ORACLE

Data Dictionary Structure (continued)

The data dictionary consists of sets of views. In many cases, a set consists of three views containing similar information and distinguished from each other by their prefixes. For example, there is a view named USER_OBJECTS, another named ALL_OBJECTS, and a third named DBA_OBJECTS.

These three views contain similar information about objects in the database, except that the scope is different. USER_OBJECTS contains information about objects that you own or created. ALL_OBJECTS contains information about all objects to which you have access. DBA_OBJECTS contains information on all objects that are owned by all users. For views that are prefixed with ALL or DBA, there is usually an additional column in the view named OWNER to identify who owns the object.

There is also a set of views that is prefixed with v$. These views are dynamic in nature and hold information about performance. Dynamic performance tables are not true tables, and they should not be accessed by most users. However, database administrators can query and create views on the tables and grant access to those views to other users. This course does not go into details about these views.

How to Use the Dictionary Views

Start with DICTIONARY. It contains the names and descriptions of the dictionary tables and views.

```
DESCRIBE DICTIONARY
```

Name	Null	Type
TABLE_NAME		VARCHAR2(30)
COMMENTS		VARCHAR2(4000)

```
SELECT *
FROM   dictionary
WHERE  table_name = 'USER_OBJECTS';
```

	TABLE_NAME	COMMENTS
1	USER_OBJECTS	Objects owned by the user

ORACLE

How to Use the Dictionary Views

To familiarize yourself with the dictionary views, you can use the dictionary view named DICTIONARY. It contains the name and short description of each dictionary view to which you have access.

You can write queries to search for information on a particular view name, or you can search the COMMENTS column for a word or phrase. In the example shown, the DICTIONARY view is described. It has two columns. The SELECT statement retrieves information about the dictionary view named USER_OBJECTS. The USER_OBJECTS view contains information about all the objects that you own.

You can write queries to search the COMMENTS column for a word or phrase. For example, the following query returns the names of all views that you are permitted to access in which the COMMENTS column contains the word *columns*:

```
SELECT table_name
FROM dictionary
WHERE LOWER(comments) LIKE '%columns';
```

Note: The names in the data dictionary are uppercase.

USER_OBJECTS and ALL_OBJECTS Views

- Use the USER_OBJECTS view to:
  - See all of the objects that are owned by you
  - Obtain a listing of all object names and types in your schema, plus the following information:
    - Date created
    - Date of last modification
    - Status (valid or invalid)
- Use the ALL_OBJECTS view to see all objects to which you have access

USER_OBJECTS View

You can query the USER_OBJECTS view to see the names and types of all the objects in your schema. There are several columns in this view:

- **OBJECT_NAME:** Name of the object
- **OBJECT_ID:** Dictionary object number of the object
- **OBJECT_TYPE:** Type of object (such as TABLE, VIEW, INDEX, SEQUENCE)
- **CREATED:** Timestamp for the creation of the object
- **LAST_DDL_TIME:** Timestamp for the last modification of the object resulting from a DDL command
- **STATUS:** Status of the object (VALID, INVALID, or N/A)
- **GENERATED:** Was the name of this object system-generated? (Y | N)

Note: This is not a complete listing of the columns. For a complete listing, see "USER_OBJECTS" in the *Oracle Database Reference*.

You can also query the ALL_OBJECTS view to see a listing of all objects to which you have access.

USER_OBJECTS View

```
SELECT object_name, object_type, created, status
FROM   user_objects
ORDER BY object_type;
```

	OBJECT_NAME	OBJECT_TYPE	CREATED	STATUS
1	REG_ID_PK	INDEX	29-OCT-08	VALID
2	DEPT_NAME_IDX	INDEX	11-NOV-08	VALID
3	DEPARTMENT_ID_PK	INDEX	11-NOV-08	VALID
4	LOC_COUNTRY_IX	INDEX	29-OCT-08	VALID
5	LOC_STATE_PROVINCE_IX	INDEX	29-OCT-08	VALID
6	LOC_CITY_IX	INDEX	29-OCT-08	VALID
7	JHIST_DEPARTMENT_IX	INDEX	29-OCT-08	VALID
8	JHIST_EMPLOYEE_IX	INDEX	29-OCT-08	VALID

...

USER_OBJECTS View (continued)

The example shows the names, types, dates of creation, and status of all objects that are owned by this user.

The OBJECT_TYPE column holds the values of either TABLE, VIEW, SEQUENCE, INDEX, PROCEDURE, FUNCTION, PACKAGE, or TRIGGER.

The STATUS column holds a value of VALID, INVALID, or N/A. While tables are always valid, the views, procedures, functions, packages, and triggers may be invalid.

The CAT View

For a simplified query and output, you can query the CAT view. This view contains only two columns: TABLE_NAME and TABLE_TYPE. It provides the names of all your INDEX, TABLE, CLUSTER, VIEW, SYNONYM, SEQUENCE, or UNDEFINED objects.

Table Information

USER_TABLES:

```
DESCRIBE user_tables
```

```
Name                          Null     Type
----------------------------- -------- -------------
TABLE_NAME                    NOT NULL VARCHAR2(30)
TABLESPACE_NAME                        VARCHAR2(30)
CLUSTER_NAME                           VARCHAR2(30)
IOT_NAME                               VARCHAR2(30)
...
```

```
SELECT table_name
FROM   user_tables;
```

	TABLE_NAME
1	REGIONS
2	LOCATIONS
3	DEPARTMENTS
4	JOBS
5	EMPLOYEES

...

USER_TABLES View

You can use the USER_TABLES view to obtain the names of all of your tables. The USER_TABLES view contains information about your tables. In addition to providing the table name, it contains detailed information on the storage.

The TABS view is a synonym of the USER_TABLES view. You can query it to see a listing of tables that you own:

```
SELECT table_name
FROM  tabs;
```

Note: For a complete listing of the columns in the USER_TABLES view, see "USER_TABLES" in the *Oracle Database Reference*.

You can also query the ALL_TABLES view to see a listing of all tables to which you have access.

Column Information

USER_TAB_COLUMNS:

```
DESCRIBE user_tab_columns
```

```
Name                           Null     Type
------------------------------ -------- --------------
TABLE_NAME                     NOT NULL VARCHAR2(30)
COLUMN_NAME                    NOT NULL VARCHAR2(30)
DATA_TYPE                               VARCHAR2(106)
DATA_TYPE_MOD                           VARCHAR2(3)
DATA_TYPE_OWNER                         VARCHAR2(30)
DATA_LENGTH                    NOT NULL NUMBER
DATA_PRECISION                          NUMBER
DATA_SCALE                              NUMBER
NULLABLE                                VARCHAR2(1)
COLUMN_ID                               NUMBER
DEFAULT_LENGTH                          NUMBER
DATA_DEFAULT                            LONG()
...
```


Column Information

You can query the USER_TAB_COLUMNS view to find detailed information about the columns in your tables. While the USER_TABLES view provides information on your table names and storage, detailed column information is found in the USER_TAB_COLUMNS view.

This view contains information such as:

- Column names
- Column data types
- Length of data types
- Precision and scale for NUMBER columns
- Whether nulls are allowed (Is there a NOT NULL constraint on the column?)
- Default value

Note: For a complete listing and description of the columns in the USER_TAB_COLUMNS view, see "USER_TAB_COLUMNS" in the *Oracle Database Reference*.

Column Information

```
SELECT column_name, data_type, data_length,
       data_precision, data_scale, nullable
FROM   user_tab_columns
WHERE  table_name = 'EMPLOYEES';
```

	COLUMN_NAME	DATA_TYPE	DATA_LENGTH	DATA_PRECISION	DATA_SCALE	NULLABLE
1	EMPLOYEE_ID	NUMBER	22	6	0	N
2	FIRST_NAME	VARCHAR2	20	(null)	(null)	Y
3	LAST_NAME	VARCHAR2	25	(null)	(null)	N
4	EMAIL	VARCHAR2	25	(null)	(null)	N
5	PHONE_NUMBER	VARCHAR2	20	(null)	(null)	Y
6	HIRE_DATE	DATE	7	(null)	(null)	N
7	JOB_ID	VARCHAR2	10	(null)	(null)	N
8	SALARY	NUMBER	22	8	2	Y
9	COMMISSION_PCT	NUMBER	22	2	2	Y
10	MANAGER_ID	NUMBER	22	6	0	Y
11	DEPARTMENT_ID	NUMBER	22	4	0	Y

ORACLE

Column Information (continued)

By querying the USER_TAB_COLUMNS table, you can find details about your columns such as the names, data types, data type lengths, null constraints, and default value for a column.

The example shown displays the columns, data types, data lengths, and null constraints for the EMPLOYEES table. Note that this information is similar to the output from the DESCRIBE command.

Constraint Information

- USER_CONSTRAINTS describes the constraint definitions on your tables.
- USER_CONS_COLUMNS describes columns that are owned by you and that are specified in constraints.

```
DESCRIBE user_constraints
```

Name	Null	Type
OWNER	NOT NULL	VARCHAR2(30)
CONSTRAINT_NAME	NOT NULL	VARCHAR2(30)
CONSTRAINT_TYPE		VARCHAR2(1)
TABLE_NAME	NOT NULL	VARCHAR2(30)
SEARCH_CONDITION		LONG()
R_OWNER		VARCHAR2(30)
R_CONSTRAINT_NAME		VARCHAR2(30)
DELETE_RULE		VARCHAR2(9)
STATUS		VARCHAR2(8)
DEFERRABLE		VARCHAR2(14)

...

Constraint Information

You can find out the names of your constraints, the type of constraint, the table name to which the constraint applies, the condition for check constraints, foreign key constraint information, deletion rule for foreign key constraints, the status, and many other types of information about your constraints.

Note: For a complete listing and description of the columns in the USER_CONSTRAINTS view, see "USER_CONSTRAINTS" in the *Oracle Database Reference*.

Constraint Information

```
SELECT constraint_name, constraint_type,
       search_condition, r_constraint_name,
       delete_rule, status
FROM   user_constraints
WHERE  table_name = 'EMPLOYEES';
```

	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION	R_CONSTRAINT_NAME	DELETE_RULE	STATUS
1	EMP_LAST_NAME_NN	C	"LAST_NAME" IS NOT NULL	(null)	(null)	ENABLED
2	EMP_EMAIL_NN	C	"EMAIL" IS NOT NULL	(null)	(null)	ENABLED
3	EMP_HIRE_DATE_NN	C	"HIRE_DATE" IS NOT NULL	(null)	(null)	ENABLED
4	EMP_JOB_NN	C	"JOB_ID" IS NOT NULL	(null)	(null)	ENABLED
5	EMP_SALARY_MIN	C	salary > 0	(null)	(null)	ENABLED
6	EMP_EMAIL_UK	U	(null)	(null)	(null)	ENABLED
7	EMP_EMP_ID_PK	P	(null)	(null)	(null)	ENABLED
8	EMP_DEPT_FK	R	(null)	DEPT_ID_PK	NO ACTION	ENABLED
9	EMP_JOB_FK	R	(null)	JOB_ID_PK	NO ACTION	ENABLED
10	EMP_MANAGER_FK	R	(null)	EMP_EMP_ID_PK	NO ACTION	ENABLED

ORACLE

USER_CONSTRAINTS: Example

In the example shown, the USER_CONSTRAINTS view is queried to find the names, types, check conditions, name of the unique constraint that the foreign key references, deletion rule for a foreign key, and status for constraints on the EMPLOYEES table.

The CONSTRAINT_TYPE can be:

- C (check constraint on a table)
- P (primary key)
- U (unique key)
- R (referential integrity)
- V (with check option, on a view)
- O (with read-only, on a view)

The DELETE_RULE can be:

- **CASCADE:** If the parent record is deleted, the child records are deleted too.
- **NO ACTION:** A parent record can be deleted only if no child records exist.

The STATUS can be:

- **ENABLED:** Constraint is active.
- **DISABLED:** Constraint is made not active.

Constraint Information

```
DESCRIBE user_cons_columns
```

```
Name                           Null     Type
------------------------------ -------- ---------------
OWNER                          NOT NULL VARCHAR2(30)
CONSTRAINT_NAME                NOT NULL VARCHAR2(30)
TABLE_NAME                     NOT NULL VARCHAR2(30)
COLUMN_NAME                             VARCHAR2(4000)
POSITION                                NUMBER
```

```
SELECT constraint_name, column_name
FROM   user_cons_columns
WHERE  table_name = 'EMPLOYEES';
```

	CONSTRAINT_NAME	COLUMN_NAME
1	EMP_EMAIL_UK	EMAIL
2	EMP_SALARY_MIN	SALARY
3	EMP_JOB_NN	JOB_ID
4	EMP_HIRE_DATE_NN	HIRE_DATE
5	EMP_EMAIL_NN	EMAIL

...

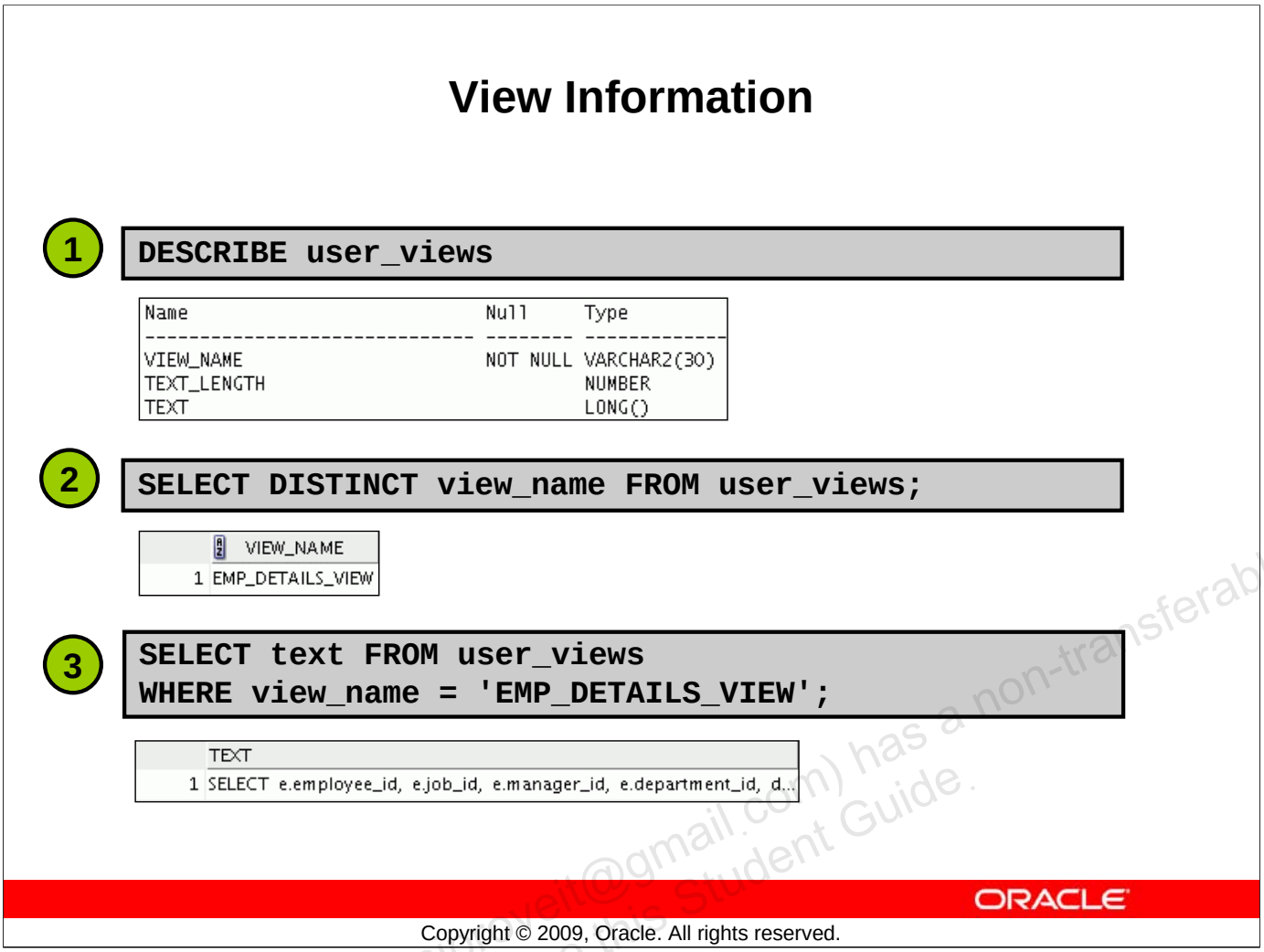
ORACLE

Querying USER_CONS_COLUMNS

To find the names of the columns to which a constraint applies, query the USER_CONS_COLUMNS dictionary view. This view tells you the name of the owner of a constraint, the name of the constraint, the table that the constraint is on, the names of the columns with the constraint, and the original position of column or attribute in the definition of the object.

Note: A constraint may apply to more than one column.

You can also write a join between the USER_CONSTRAINTS and USER_CONS_COLUMNS to create customized output from both tables.



Views in the Data Dictionary

After your view is created, you can query the data dictionary view called `USER_VIEWS` to see the name of the view and the view definition. The text of the `SELECT` statement that constitutes your view is stored in a `LONG` column. The `LENGTH` column is the number of characters in the `SELECT` statement. By default, when you select from a `LONG` column, only the first 80 characters of the column's value are displayed. To see more than 80 characters in SQL*Plus, use the command `SET LONG`:

```
SET LONG 1000
```

In the examples in the slide:

1. The `USER_VIEWS` columns are displayed. Note that this is a partial listing.
2. The names of your views are retrieved.
3. The `SELECT` statement for the `EMP_DETAILS_VIEW` is displayed from the dictionary.

Data Access Using Views

When you access data using a view, the Oracle server performs the following operations:

- It retrieves the view definition from the data dictionary table `USER_VIEWS`.
- It checks access privileges for the view base table.
- It converts the view query into an equivalent operation on the underlying base table or tables. In other words, data is retrieved from, or an update is made to, the base tables.

Sequence Information

```
DESCRIBE user_sequences
```

Name	Null	Type
SEQUENCE_NAME	NOT NULL	VARCHAR2(30)
MIN_VALUE		NUMBER
MAX_VALUE		NUMBER
INCREMENT_BY	NOT NULL	NUMBER
CYCLE_FLAG		VARCHAR2(1)
ORDER_FLAG		VARCHAR2(1)
CACHE_SIZE	NOT NULL	NUMBER
LAST_NUMBER	NOT NULL	NUMBER

ORACLE

USER_SEQUENCES View

The USER_SEQUENCES view describes all sequences that are owned by you. When you create the sequence, you specify criteria that are stored in the USER_SEQUENCES view. The columns in this view are:

- **SEQUENCE_NAME:** Name of the sequence
- **MIN_VALUE:** Minimum value of the sequence
- **MAX_VALUE:** Maximum value of the sequence
- **INCREMENT_BY:** Value by which sequence is incremented
- **CYCLE_FLAG:** Does sequence wrap around on reaching limit?
- **ORDER_FLAG:** Are sequence numbers generated in order?
- **CACHE_SIZE:** Number of sequence numbers to cache
- **LAST_NUMBER:** Last sequence number written to disk. If a sequence uses caching, the number written to disk is the last number placed in the sequence cache. This number is likely to be greater than the last sequence number that was used.

Sequence Information

- Verify your sequence values in the USER_SEQUENCES data dictionary table.

```
SELECT   sequence_name, min_value, max_value,
         increment_by, last_number
FROM     user_sequences;
```

	SEQUENCE_NAME	MIN_VALUE	MAX_VALUE	INCREMENT_BY	LAST_NUMBER
1	LOCATIONS_SEQ	1	9900	100	3300
2	DEPARTMENTS_SEQ	1	9990	10	280
3	EMPLOYEES_SEQ	1	999999999999999999999999999	1	207
4	DEPT_ID_SEQ	1	1000	10	400

- The LAST_NUMBER column displays the next available sequence number if NOCACHE is specified.

ORACLE

Confirming Sequences

After creating your sequence, it is documented in the data dictionary. Because a sequence is a database object, you can identify it in the USER_OBJECTS data dictionary table.

You can also confirm the settings of the sequence by selecting from the USER_SEQUENCES data dictionary view.

Viewing the Next Available Sequence Value Without Incrementing It

If the sequence was created with NOCACHE, it is possible to view the next available sequence value without incrementing it by querying the USER_SEQUENCES table.

Synonym Information

```
DESCRIBE user_synonyms
```

Name	Null	Type
SYNONYM_NAME	NOT NULL	VARCHAR2(30)
TABLE_OWNER		VARCHAR2(30)
TABLE_NAME	NOT NULL	VARCHAR2(30)
DB_LINK		VARCHAR2(128)

```
SELECT *
FROM   user_synonyms;
```

	SYNONYM_NAME	TABLE_OWNER	TABLE_NAME	DB_LINK
1	EMP	ORA1	EMPLOYEES	(null)

ORACLE

USER_SYNONYMS View

The USER_SYNONYMS dictionary view describes private synonyms (synonyms that are owned by you).

You can query this view to find your synonyms. You can query ALL_SYNONYMS to find out the name of all of the synonyms that are available to you and the objects on which these synonyms apply.

The columns in this view are:

- **SYNONYM_NAME:** Name of the synonym
- **TABLE_OWNER:** Owner of the object that is referenced by the synonym
- **TABLE_NAME:** Name of the table or view that is referenced by the synonym
- **DB_LINK:** Name of the database link reference (if any)

Adding Comments to a Table

- You can add comments to a table or column by using the COMMENT statement:

```
COMMENT ON TABLE employees
IS 'Employee Information';
COMMENT ON succeeded.
```

- Comments can be viewed through the data dictionary views:
  - ALL_COL_COMMENTS
  - USER_COL_COMMENTS
  - ALL_TAB_COMMENTS
  - USER_TAB_COMMENTS

ORACLE

Adding Comments to a Table

You can add a comment of up to 4,000 bytes about a column, table, view, or snapshot by using the COMMENT statement. The comment is stored in the data dictionary and can be viewed in one of the following data dictionary views in the COMMENTS column:

- ALL_COL_COMMENTS
- USER_COL_COMMENTS
- ALL_TAB_COMMENTS
- USER_TAB_COMMENTS

Syntax

```
COMMENT ON TABLE table | COLUMN table.column
    IS 'text';
```

In the syntax:

table	Is the name of the table
column	Is the name of the column in a table
text	Is the text of the comment

You can drop a comment from the database by setting it to empty string (''):

```
COMMENT ON TABLE  employees IS ' ';
```

Summary

In this lesson, you should have learned how to find information about your objects by using the following dictionary views:

- DICTIONARY
- USER_OBJECTS
- USER_TABLES
- USER_TAB_COLUMNS
- USER_CONSTRAINTS
- USER_CONS_COLUMNS
- USER_VIEWS
- USER_SEQUENCES
- USER_TAB_SYNONYMS

ORACLE

Summary

In this lesson, you learned about some of the dictionary views that are available to you. You can use these dictionary views to find information about your tables, constraints, views, sequences, and synonyms.

Practice 11: Overview

This practice covers the following topics:

- Querying the dictionary views for table and column information
- Querying the dictionary views for constraint information
- Querying the dictionary views for view information
- Querying the dictionary views for sequence information
- Querying the dictionary views for synonym information
- Adding a comment to a table and querying the dictionary views for comment information

ORACLE

Practice 11: Overview

In this practice, you query the dictionary views to find information about the objects in your schema.

Practice 11

1. For a specified table, create a script that reports the column names, data types, and lengths of the data types, as well as whether nulls are allowed. Prompt the user to enter the table name. Give appropriate aliases to the `DATA_PRECISION` and `DATA_SCALE` columns. Save this script in a file named `lab_11_01.sql`.

   For example, if the user enters `DEPARTMENTS`, the following output results:

	COLUMN_NAME	DATA_TYPE	DATA_LENGTH	PRECISION	SCALE	NULLABLE
1	DEPARTMENT_ID	NUMBER	22	4	0	N
2	DEPARTMENT_NAME	VARCHAR2	30	(null)	(null)	N
3	MANAGER_ID	NUMBER	22	6	0	Y
4	LOCATION_ID	NUMBER	22	4	0	Y

2. Create a script that reports the column name, constraint name, constraint type, search condition, and status for a specified table. You must join the `USER_CONSTRAINTS` and `USER_CONS_COLUMNS` tables to obtain all this information. Prompt the user to enter the table name. Save the script in a file named `lab_11_02.sql`.

   For example, if the user enters `DEPARTMENTS`, the following output results:

	COLUMN_NAME	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION	STATUS
1	DEPARTMENT_NAME	DEPT_NAME_NN	C	"DEPARTMENT_NAME" IS NOT NULL	ENABLED
2	DEPARTMENT_ID	DEPT_ID_PK	P	(null)	ENABLED
3	LOCATION_ID	DEPT_LOC_FK	R	(null)	ENABLED
4	MANAGER_ID	DEPT_MGR_FK	R	(null)	ENABLED

3. Add a comment to the `DEPARTMENTS` table. Then query the `USER_TAB_COMMENTS` view to verify that the comment is present.

	COMMENTS
1	Company department information including name, code, and location.

4. Find the names of all synonyms that are in your schema.

	SYNONYM_NAME	TABLE_OWNER	TABLE_NAME	DB_LINK
1	EMP	ORA1	EMPLOYEES	(null)

Practice 11 (continued)

5. You need to determine the names and definitions of all the views in your schema. Create a report that retrieves the view information: the view name and text from the `USER_VIEWS` data dictionary view.
   Note: Another view already exists. The `EMP_DETAILS_VIEW` was created as part of your schema. Also, if you completed practice 10, you see the `DEPT50` view.
   Note: To see more contents of a `LONG` column, use the command `SET LONG` *n*, where *n* is the value of the number of characters of the `LONG` column that you want to see.

	VIEW_NAME	TEXT
1	EMP_DETAILS_VIEW	SELECT e.employee_id, e.job_id, e.ma...
2	EMPLOYEES_VU	SELECT employee_id, last_name emplo...

6. Find the names of your sequences. Write a query in a script to display the following information about your sequences: sequence name, maximum value, increment size, and last number. Name the script `lab_11_06.sql`. Run the statement in your script.

	SEQUENCE_NAME	MAX_VALUE	INCREMENT_BY	LAST_NUMBER
1	LOCATIONS_SEQ	9900	100	3300
2	DEPARTMENTS_SEQ	9990	10	280
3	EMPLOYEES_SEQ	999999999999999999999999	1	207
4	DEPT_ID_SEQ	1000	10	400