

10

Using DDL Statements to Create and Manage Tables

Objectives

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

- Categorize the main database objects
- Review the table structure
- List the data types that are available for columns
- Create a simple table
- Explain how constraints are created at the time of table creation
- Describe how schema objects work

Database Objects

Object	Description
Table	Basic unit of storage; composed of rows
View	Logically represents subsets of data from one or more tables
Sequence	Generates numeric values
Index	Improves the performance of some queries
Synonym	Gives alternative name to an object

Naming Rules

Table names and column names must:

- Begin with a letter
- Be 1–30 characters long
- Contain only A–Z, a–z, 0–9, _, \$, and #
- Not duplicate the name of another object owned by the same user
- Not be an Oracle server–reserved word

CREATE TABLE Statement

- You must have:
 - The CREATE TABLE privilege
 - A storage area

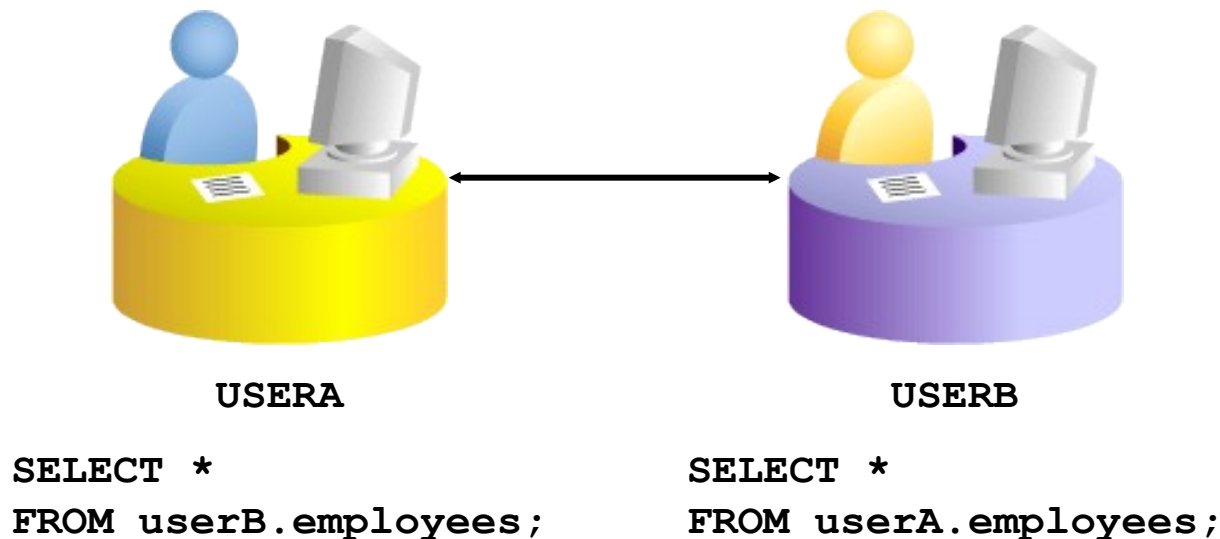
```
CREATE TABLE [schema.]table  
      (column datatype [DEFAULT expr][, ...]);
```

- You specify:
 - The table name
 - The column name, column data type, and column size



Referencing Another User's Tables

- Tables belonging to other users are not in the user's schema.
- You should use the owner's name as a prefix to those tables.



DEFAULT Option

- Specify a default value for a column during an insert.

```
... hire_date DATE DEFAULT SYSDATE, ...
```

- Literal values, expressions, or SQL functions are legal values.
- Another column's name or a pseudocolumn are illegal values.
- The default data type must match the column data type.

```
CREATE TABLE hire_dates  
  (id          NUMBER(8),  
   hire_date DATE DEFAULT SYSDATE);
```

```
CREATE TABLE succeeded.
```

Creating Tables

- Create the table:

```
CREATE TABLE dept
      (deptno      NUMBER(2) ,
       dname       VARCHAR2(14) ,
       loc         VARCHAR2(13) ,
       create_date DATE DEFAULT SYSDATE) ;
```

```
CREATE TABLE succeeded.
```

- Confirm table creation:

```
DESCRIBE dept
```

```
DESCRIBE dept
Name                               Null    Type
-----
DEPTNO                             NUMBER(2)
DNAME                             VARCHAR2(14)
LOC                               VARCHAR2(13)
CREATE_DATE                        DATE
4 rows selected
```


Data Types

Data Type	Description
VARCHAR2 (<i>size</i>)	Variable-length character data
CHAR (<i>size</i>)	Fixed-length character data
NUMBER (<i>p</i> , <i>s</i>)	Variable-length numeric data
DATE	Date and time values
LONG	Variable-length character data (up to 2 GB)
CLOB	Character data (up to 4 GB)
RAW and LONG RAW	Raw binary data
BLOB	Binary data (up to 4 GB)
BFILE	Binary data stored in an external file (up to 4 GB)
ROWID	A base-64 number system representing the unique address of a row in its table

Datetime Data Types

You can use several datetime data types:

Data Type	Description
TIMESTAMP	Date with fractional seconds
INTERVAL YEAR TO MONTH	Stored as an interval of years and months
INTERVAL DAY TO SECOND	Stored as an interval of days, hours, minutes, and seconds



Including Constraints

- Constraints enforce rules at the table level.
- Constraints prevent the deletion of a table if there are dependencies.
- The following constraint types are valid:
 - NOT NULL
 - UNIQUE
 - PRIMARY KEY
 - FOREIGN KEY
 - CHECK



Constraint Guidelines

- You can name a constraint, or the Oracle server generates a name by using the `SYS_Cn` format.
- Create a constraint at either of the following times:
 - At the same time as the creation of the table
 - After the creation of the table
- Define a constraint at the column or table level.
- View a constraint in the data dictionary.

Defining Constraints

- Syntax:

```
CREATE TABLE [schema.]table
    (column datatype [DEFAULT expr]
     [column_constraint],
     ...
     [table_constraint][, ...]);
```

- Column-level constraint syntax:

```
column [CONSTRAINT constraint_name] constraint_type,
```

- Table-level constraint syntax:

```
column, ...
    [CONSTRAINT constraint_name] constraint_type
    (column, ...),
```

Defining Constraints

- Example of a column-level constraint:

```
CREATE TABLE employees (  
  employee_id  NUMBER(6)  
    CONSTRAINT emp_emp_id_pk PRIMARY KEY,  
  first_name   VARCHAR2(20) ,  
  ... ) ;
```

1

- Example of a table-level constraint:

```
CREATE TABLE employees (  
  employee_id  NUMBER(6) ,  
  first_name   VARCHAR2(20) ,  
  ...  
  job_id       VARCHAR2(10) NOT NULL ,  
  CONSTRAINT emp_emp_id_pk  
    PRIMARY KEY (EMPLOYEE_ID) ) ;
```

2

NOT NULL Constraint

Ensures that null values are not permitted for the column:

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	SALARY	COMMISSION_PCT	DEPARTMENT_ID	EMAIL	PHONE_NUMBER	HIRE_DATE
100	Steven	King	24000	(null)	90	SKING	515.123.4567	17-JUN-87
101	Neena	Kochhar	17000	(null)	90	NKOCHHAR	515.123.4568	21-SEP-89
102	Lex	De Haan	17000	(null)	90	LDEHAAN	515.123.4569	13-JAN-93
103	Alexander	Hunold	9000	(null)	60	AHUNOLD	590.423.4567	03-JAN-90
104	Bruce	Ernst	6000	(null)	60	BERNST	590.423.4568	21-MAY-91
107	Diana	Lorentz	4200	(null)	60	DLORENTZ	590.423.5567	07-FEB-99
124	Kevin	Mourgos	5800	(null)	50	KMOURGOS	650.123.5234	16-NOV-99
141	Trenna	Rajs	3500	(null)	50	TRAJS	650.121.8009	17-OCT-95
142	Curtis	Davies	3100	(null)	50	CDAVIES	650.121.2994	29-JAN-97
143	Randall	Matos	2600	(null)	50	RMATOS	650.121.2874	15-MAR-98
144	Peter	Vargas	2500	(null)	50	PVARGAS	650.121.2004	09-JUL-98
149	Eleni	Zlotkey	10500	0.2	80	EZLOTKEY	011.44.1344.429018	29-JAN-00
174	Ellen	Abel	11000	0.3	80	EABEL	011.44.1644.429267	11-MAY-96
176	Jonathon	Taylor	8600	0.2	80	JTAYLOR	011.44.1644.429265	24-MAR-98
178	Kimberely	Grant	7000	0.15	(null)	KGRANT	011.44.1644.429263	24-MAY-99
200	Jennifer	Whalen	4400	(null)	10	JWHALEN	515.123.4444	17-SEP-87
201	Michael	Hartstein	13000	(null)	20	MHARTSTE	515.123.5555	17-FEB-96
202	Pat	Fay	6000	(null)	20	PFAY	603.123.6666	17-AUG-97
205	Shelley	Higgins	12000	(null)	110	SHIGGINS	515.123.8080	07-JUN-94
206	William	Gietz	8300	(null)	110	WGIEZT	515.123.8181	07-JUN-94

NOT NULL constraint
(Primary Key enforces
NOT NULL constraint.)


NOT NULL
constraint

Absence of NOT NULL
constraint (Any row can
contain a null value for this
column)

UNIQUE Constraint

EMPLOYEES

UNIQUE constraint



	EMPLOYEE_ID	LAST_NAME	EMAIL
1	100	King	SKING
2	101	Kochhar	NKOCHHAR
3	102	De Haan	LDEHAAN
4	103	Hunold	AHUNOLD
5	104	Ernst	BERNST
6	107	Lorentz	DLORENTZ

...



INSERT INTO

208	SMITH	JSMITH
-----	-------	--------

← Allowed

209	SMITH	JSMITH
-----	-------	--------

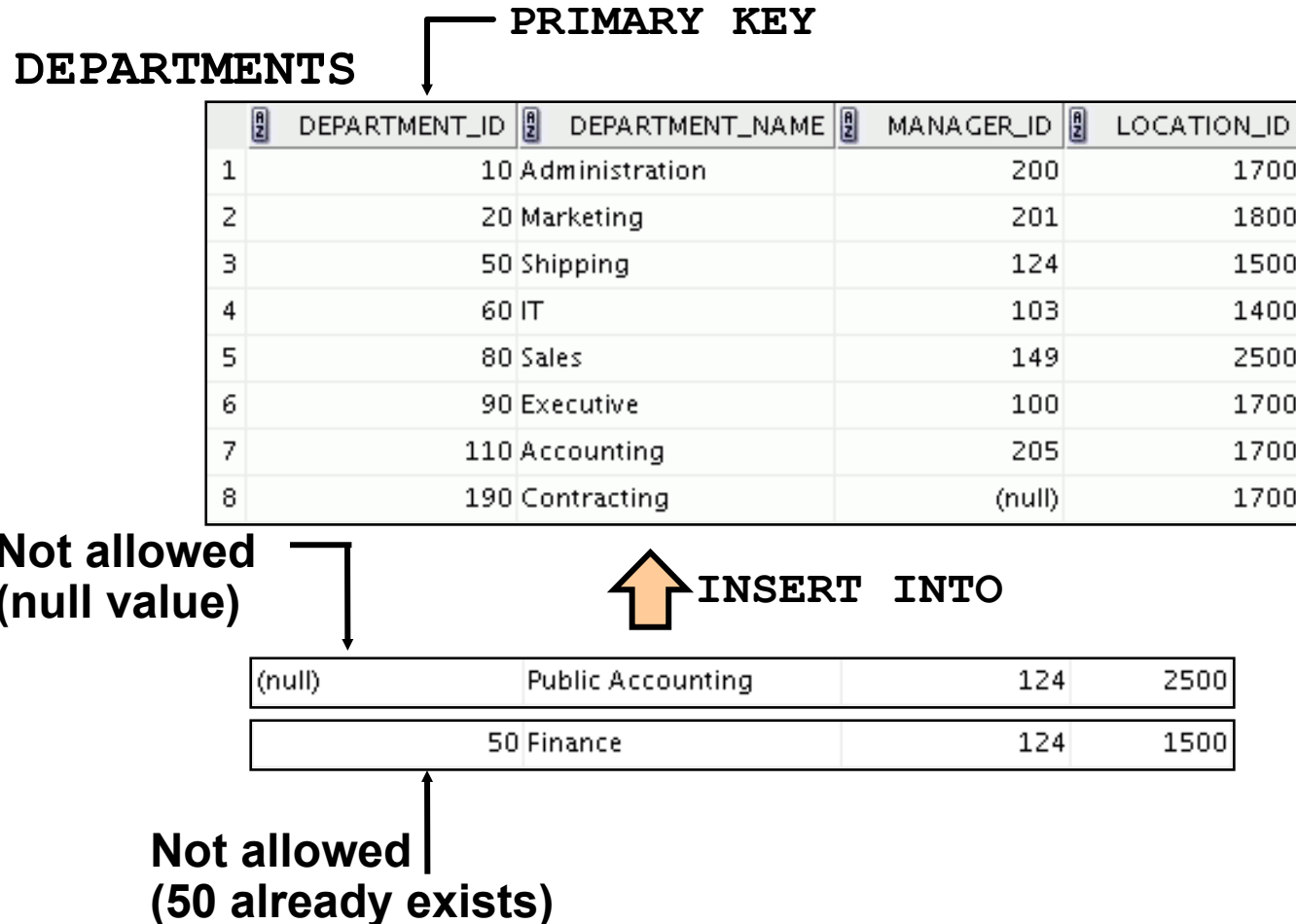
← Not allowed: already exists

UNIQUE Constraint

Defined at either the table level or the column level:

```
CREATE TABLE employees(  
    employee_id      NUMBER(6) ,  
    last_name        VARCHAR2(25) NOT NULL ,  
    email            VARCHAR2(25) ,  
    salary           NUMBER(8,2) ,  
    commission_pct   NUMBER(2,2) ,  
    hire_date        DATE NOT NULL ,  
    ...  
    CONSTRAINT emp_email_uk UNIQUE(email));
```

PRIMARY KEY Constraint



FOREIGN KEY Constraint

DEPARTMENTS

PRIMARY KEY →

	DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
1	10	Administration	200	1700
2	20	Marketing	201	1800
3	50	Shipping	124	1500
4	60	IT	103	1400
5	80	Sales	149	2500

...

EMPLOYEES

	EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
1	200	Whalen	10
2	201	Hartstein	20
3	202	Fay	20
4	205	Higgins	110
5	206	Gietz	110

FOREIGN KEY

...



INSERT INTO

200	Ford	9
200	Ford	60

**Not allowed
(9 does not
exist)**

Allowed

FOREIGN KEY Constraint

Defined at either the table level or the column level:

```
CREATE TABLE employees(  
    employee_id      NUMBER(6) ,  
    last_name        VARCHAR2(25) NOT NULL,  
    email            VARCHAR2(25) ,  
    salary            NUMBER(8,2) ,  
    commission_pct   NUMBER(2,2) ,  
    hire_date        DATE NOT NULL,  
    ...  
    department_id    NUMBER(4) ,  
    CONSTRAINT emp_dept_fk FOREIGN KEY (department_id)  
        REFERENCES departments(department_id) ,  
    CONSTRAINT emp_email_uk UNIQUE(email)) ;
```

FOREIGN KEY Constraint: Keywords

- FOREIGN KEY: Defines the column in the child table at the table-constraint level
- REFERENCES: Identifies the table and column in the parent table
- ON DELETE CASCADE: Deletes the dependent rows in the child table when a row in the parent table is deleted
- ON DELETE SET NULL: Converts dependent foreign key values to null

CHECK Constraint

- Defines a condition that each row must satisfy
- The following expressions are not allowed:
 - References to CURRVAL, NEXTVAL, LEVEL, and ROWNUM pseudocolumns
 - Calls to SYSDATE, UID, USER, and USERENV functions
 - Queries that refer to other values in other rows

```
..., salary  NUMBER(2)  
      CONSTRAINT emp_salary_min  
      CHECK (salary > 0),...
```

CREATE TABLE: Example

```
CREATE TABLE employees
( employee_id      NUMBER(6)
  CONSTRAINT emp_employee_id PRIMARY KEY
, first_name      VARCHAR2(20)
, last_name       VARCHAR2(25)
  CONSTRAINT emp_last_name_nn NOT NULL
, email           VARCHAR2(25)
  CONSTRAINT emp_email_nn    NOT NULL
  CONSTRAINT emp_email_uk    UNIQUE
, phone_number    VARCHAR2(20)
, hire_date       DATE
  CONSTRAINT emp_hire_date_nn NOT NULL
, job_id          VARCHAR2(10)
  CONSTRAINT emp_job_nn      NOT NULL
, salary          NUMBER(8,2)
  CONSTRAINT emp_salary_ck   CHECK (salary>0)
, commission_pct  NUMBER(2,2)
, manager_id      NUMBER(6)
  CONSTRAINT emp_manager_fk REFERENCES
    employees (employee_id)
, department_id   NUMBER(4)
  CONSTRAINT emp_dept_fk     REFERENCES
    departments (department_id));
```

Violating Constraints

```
UPDATE employees
SET   department_id = 55
WHERE department_id = 110;
```

Error starting at line 1 in command:

UPDATE employees

SET department_id = 55

WHERE department_id = 110

Error report:

SQL Error: ORA-02291: integrity constraint (ORA1.EMP_DEPT_FK) violated - parent key not found

02291. 00000 - "integrity constraint (%s.%s) violated - parent key not found"

*Cause: A foreign key value has no matching primary key value.

Department 55 does not exist.

Violating Constraints

You cannot delete a row that contains a primary key that is used as a foreign key in another table.

```
DELETE FROM departments
WHERE department_id = 60;
```

Error starting at line 1 in command: DELETE FROM departments WHERE department_id = 60	
Error report: SQL Error: ORA-02292: integrity constraint (ORA1.JHIST_DEPT_FK) violated - child record found 02292. 00000 - "integrity constraint (%s.%s) violated - child record found" *Cause: attempted to delete a parent key value that had a foreign dependency. *Action: delete dependencies first then parent or disable constraint.	

Creating a Table Using a Subquery

- Create a table and insert rows by combining the `CREATE TABLE` statement and the `AS subquery` option.

```
CREATE TABLE table  
            [ (column, column... ) ]  
AS subquery;
```

- Match the number of specified columns to the number of subquery columns.
- Define columns with column names and default values.

Creating a Table Using a Subquery

```
CREATE TABLE dept80
AS
SELECT  employee_id, last_name,
        salary*12 ANNSAL,
        hire_date
FROM    employees
WHERE   department_id = 80;
```

CREATE TABLE succeeded.

```
DESCRIBE dept80
```

Name	Null	Type
-----	-----	-----
EMPLOYEE_ID		NUMBER(6)
LAST_NAME	NOT NULL	VARCHAR2(25)
ANNSAL		NUMBER
HIRE_DATE	NOT NULL	DATE

ALTER TABLE Statement

Use the `ALTER TABLE` statement to:

- Add a new column
- Modify an existing column definition
- Define a default value for the new column
- Drop a column
- Rename a column
- Change table to read-only status

Read-Only Tables

You can use the `ALTER TABLE` syntax to:

- Put a table into read-only mode, which prevents DDL or DML changes during table maintenance
- Put the table back into read/write mode

```
ALTER TABLE employees READ ONLY;

-- perform table maintenance and then
-- return table back to read/write mode

ALTER TABLE employees READ WRITE;
```

Dropping a Table

- Moves a table to the recycle bin
- Removes the table and all its data entirely if the `PURGE` clause is specified
- Invalidates dependent objects and removes object privileges on the table

```
DROP TABLE dept80;
```

```
DROP TABLE dept80 succeeded.
```

Quiz

You can use constraints to do the following:

1. Enforce rules on the data in a table whenever a row is inserted, updated, or deleted.
2. Prevent the deletion of a table.
3. Prevent the creation of a table.
4. Prevent the creation of data in a table.

Summary

In this lesson, you should have learned how to use the `CREATE TABLE` statement to create a table and include constraints:

- Categorize the main database objects
- Review the table structure
- List the data types that are available for columns
- Create a simple table
- Explain how constraints are created at the time of table creation
- Describe how schema objects work