Including Constraints
Objectives

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

• Describe constraints
• Create and maintain constraints
What are 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

• Name a constraint or the Oracle server generates a name by using the \texttt{SYS\_Cn} format.

• Create a constraint either:
  – At the same time as the table is created, or
  – After the table has been created

• Define a constraint at the column or table level.

• View a constraint in the data dictionary.
Defining Constraints

CREATE TABLE [schema.]table
  (column datatype [DEFAULT expr] [column_constraint], ...
  [table_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));
Defining Constraints

- Column constraint level

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

- Table constraint level

```
column,...
[CONSTRAINT constraint_name] constraint_type
(column,...),
```
## The NOT NULL Constraint

Ensures that null values are not permitted for the column:

<table>
<thead>
<tr>
<th>EMPLOYEE_ID</th>
<th>LAST_NAME</th>
<th>EMAIL</th>
<th>PHONE_NUMBER</th>
<th>HIRE_DATE</th>
<th>JOB_ID</th>
<th>SALARY</th>
<th>DEPARTMENT_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>King</td>
<td>SKING</td>
<td>515.123.4567</td>
<td>17-JUN-87</td>
<td>AD_PRES</td>
<td>24000</td>
<td>90</td>
</tr>
<tr>
<td>101</td>
<td>Kochhar</td>
<td>NKOCHHAR</td>
<td>515.123.4568</td>
<td>21-SEP-89</td>
<td>AD_VP</td>
<td>17000</td>
<td>90</td>
</tr>
<tr>
<td>102</td>
<td>De Haan</td>
<td>LDEHAAN</td>
<td>515.123.4569</td>
<td>13-JAN-93</td>
<td>AD_VP</td>
<td>17000</td>
<td>90</td>
</tr>
<tr>
<td>103</td>
<td>Hunold</td>
<td>AHUNOLD</td>
<td>590.423.4567</td>
<td>03-JAN-90</td>
<td>IT_PROG</td>
<td>9000</td>
<td>60</td>
</tr>
<tr>
<td>104</td>
<td>Ernst</td>
<td>BERNST</td>
<td>590.423.4568</td>
<td>21-MAY-91</td>
<td>IT_PROG</td>
<td>6000</td>
<td>60</td>
</tr>
<tr>
<td>178</td>
<td>Grant</td>
<td>KGRANT</td>
<td>011.44.1644.429263</td>
<td>24-MAY-99</td>
<td>SA_REP</td>
<td>7000</td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>Whalen</td>
<td>JWHALEN</td>
<td>515.123.4444</td>
<td>17-SEP-87</td>
<td>AD_ASST</td>
<td>4400</td>
<td>10</td>
</tr>
</tbody>
</table>

20 rows selected.

**NOT NULL constraint**
(No row can contain a null value for this column.)

**NOT NULL constraint**

**Absence of NOT NULL constraint**
(Any row can contain null for this column.)
The NOT NULL Constraint

Is defined at the column level:

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

System named

User named
The **UNIQUE** Constraint

<table>
<thead>
<tr>
<th>EMPLOYEE_ID</th>
<th>LAST_NAME</th>
<th>EMAIL</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>King</td>
<td>SKING</td>
</tr>
<tr>
<td>101</td>
<td>Kochhar</td>
<td>NKochhar</td>
</tr>
<tr>
<td>102</td>
<td>De Haan</td>
<td>LDEHAAN</td>
</tr>
<tr>
<td>103</td>
<td>Hunold</td>
<td>AHUNOLD</td>
</tr>
<tr>
<td>104</td>
<td>Ernst</td>
<td>BERNST</td>
</tr>
</tbody>
</table>

...  

**UNIQUE constraint**

**INSERT INTO**

<table>
<thead>
<tr>
<th>EMPLOYEE_ID</th>
<th>LAST_NAME</th>
<th>EMAIL</th>
</tr>
</thead>
<tbody>
<tr>
<td>208</td>
<td>Smith</td>
<td>JSMITH</td>
</tr>
<tr>
<td>209</td>
<td>Smith</td>
<td>JSMITH</td>
</tr>
</tbody>
</table>

- Allowed
- Not allowed: already exists
The **UNIQUE** Constraint

Defined at either the table level or the column level:

```sql
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));
```
The PRIMARY KEY Constraint

DEPARTMENTS

<table>
<thead>
<tr>
<th>DEPARTMENT_ID</th>
<th>DEPARTMENT_NAME</th>
<th>MANAGER_ID</th>
<th>LOCATION_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>Administration</td>
<td>200</td>
<td>1700</td>
</tr>
<tr>
<td>20</td>
<td>Marketing</td>
<td>201</td>
<td>1800</td>
</tr>
<tr>
<td>50</td>
<td>Shipping</td>
<td>124</td>
<td>1500</td>
</tr>
<tr>
<td>60</td>
<td>IT</td>
<td>103</td>
<td>1400</td>
</tr>
<tr>
<td>80</td>
<td>Sales</td>
<td>149</td>
<td>2500</td>
</tr>
</tbody>
</table>

... Not allowed (Null value)

Not allowed (50 already exists)

INSERT INTO

<table>
<thead>
<tr>
<th></th>
<th>DEPARTMENT_NAME</th>
<th>MANAGER_ID</th>
<th>LOCATION_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public Accounting</td>
<td></td>
<td></td>
<td>1400</td>
</tr>
<tr>
<td>50</td>
<td>Finance</td>
<td>124</td>
<td>1500</td>
</tr>
</tbody>
</table>
The **PRIMARY KEY** Constraint

Defined at either the table level or the column level:

```sql
CREATE TABLE departments(
    department_id        NUMBER(4),
    department_name      VARCHAR2(30),
    CONSTRAINT dept_name_nn NOT NULL,
    manager_id           NUMBER(6),
    location_id          NUMBER(4),
    CONSTRAINT dept_id_pk PRIMARY KEY(department_id));
```
### The FOREIGN KEY Constraint

**DEPARTMENTS**

<table>
<thead>
<tr>
<th>DEPARTMENT_ID</th>
<th>DEPARTMENT_NAME</th>
<th>MANAGER_ID</th>
<th>LOCATION_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>Administration</td>
<td>200</td>
<td>1700</td>
</tr>
<tr>
<td>20</td>
<td>Marketing</td>
<td>201</td>
<td>1800</td>
</tr>
<tr>
<td>50</td>
<td>Shipping</td>
<td>124</td>
<td>1500</td>
</tr>
<tr>
<td>60</td>
<td>IT</td>
<td>103</td>
<td>1400</td>
</tr>
<tr>
<td>80</td>
<td>Sales</td>
<td>149</td>
<td>2500</td>
</tr>
</tbody>
</table>

**EMPLOYEES**

<table>
<thead>
<tr>
<th>EMPLOYEE_ID</th>
<th>LAST_NAME</th>
<th>DEPARTMENT_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>King</td>
<td>90</td>
</tr>
<tr>
<td>101</td>
<td>Kochhar</td>
<td>90</td>
</tr>
<tr>
<td>102</td>
<td>De Haan</td>
<td>90</td>
</tr>
<tr>
<td>103</td>
<td>Hunold</td>
<td>60</td>
</tr>
<tr>
<td>104</td>
<td>Ernst</td>
<td>60</td>
</tr>
<tr>
<td>107</td>
<td>Lorentz</td>
<td>60</td>
</tr>
</tbody>
</table>

**Notes:**
- **PRIMARY KEY**
- **FOREIGN KEY**
- **INSERT INTO**
- Not allowed (9 does not exist)
- Allowed
The FOREIGN KEY Constraint

Defined at either the table level or the column level:

```sql
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
The **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

```sql
..., salary  NUMBER(2)
  CONSTRAINT emp_salary_min
    CHECK (salary > 0),...
```
Adding a Constraint Syntax

Use the ALTER TABLE statement to:

• Add or drop a constraint, but not modify its structure
• Enable or disable constraints
• Add a NOT NULL constraint by using the MODIFY clause

```sql
ALTER TABLE table
ADD [CONSTRAINT constraint] type (column);
```
Adding a Constraint

Add a FOREIGN KEY constraint to the EMPLOYEES table indicating that a manager must already exist as a valid employee in the EMPLOYEES table.

```
ALTER TABLE employees
ADD CONSTRAINT emp_manager_fk
    FOREIGN KEY(manager_id)
    REFERENCES employees(employee_id);
```

Table altered.
Dropping a Constraint

- Remove the manager constraint from the EMPLOYEES table.
  
  ```sql
  ALTER TABLE employees
  DROP CONSTRAINT emp_manager_fk;
  Table altered.
  ```

- Remove the PRIMARY KEY constraint on the DEPARTMENTS table and drop the associated FOREIGN KEY constraint on the EMPLOYEES.DEPARTMENT_ID column.
  
  ```sql
  ALTER TABLE departments
  DROP PRIMARY KEY CASCADE;
  Table altered.
  ```
Disabling Constraints

- Execute the `DISABLE` clause of the `ALTER TABLE` statement to deactivate an integrity constraint.
- Apply the `CASCADE` option to disable dependent integrity constraints.

```
ALTER TABLE employees
DISABLE CONSTRAINT emp_emp_id_pk CASCADE;
```

Table altered.
Enabling Constraints

- Activate an integrity constraint currently disabled in the table definition by using the `ENABLE` clause.

```sql
ALTER TABLE employees
ENABLE CONSTRAINT emp_emp_id_pk;
Table altered.
```

- A **UNIQUE** or **PRIMARY KEY** index is automatically created if you enable a **UNIQUE** key or **PRIMARY KEY** constraint.
Cascading Constraints

- The `CASCADE CONSTRAINTS` clause is used along with the `DROP COLUMN` clause.
- The `CASCADE CONSTRAINTS` clause drops all referential integrity constraints that refer to the primary and unique keys defined on the dropped columns.
- The `CASCADE CONSTRAINTS` clause also drops all multicolumn constraints defined on the dropped columns.
Cascading Constraints

Example:

```
ALTER TABLE test1
DROP (pk) CASCADE CONSTRAINTS;
Table altered.
```

```
ALTER TABLE test1
DROP (pk, fk, col1) CASCADE CONSTRAINTS;
Table altered.
```
Viewing Constraints

Query the `USER_CONSTRAINTS` table to view all constraint definitions and names.

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

<table>
<thead>
<tr>
<th>CONSTRAINT_NAME</th>
<th>C</th>
<th>SEARCH_CONDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>EMP_LAST_NAME_NN</td>
<td>C</td>
<td>&quot;LAST_NAME&quot; IS NOT NULL</td>
</tr>
<tr>
<td>EMP_EMAIL_NN</td>
<td>C</td>
<td>&quot;EMAIL&quot; IS NOT NULL</td>
</tr>
<tr>
<td>EMP_HIRE_DATE_NN</td>
<td>C</td>
<td>&quot;HIRE_DATE&quot; IS NOT NULL</td>
</tr>
<tr>
<td>EMP_JOB_NN</td>
<td>C</td>
<td>&quot;JOB_ID&quot; IS NOT NULL</td>
</tr>
<tr>
<td>EMP_SALARY_MIN</td>
<td>C</td>
<td>salary &gt; 0</td>
</tr>
<tr>
<td>EMP_EMAIL_UK</td>
<td>U</td>
<td></td>
</tr>
</tbody>
</table>
Viewing the Columns Associated with Constraints

View the columns associated with the constraint names in the USER_CONS_COLUMNS view.

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

<table>
<thead>
<tr>
<th>CONSTRAINT_NAME</th>
<th>COLUMN_NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>EMP_DEPT_FK</td>
<td>DEPARTMENT_ID</td>
</tr>
<tr>
<td>EMP_EMAIL_NN</td>
<td>EMAIL</td>
</tr>
<tr>
<td>EMP_EMAIL_UK</td>
<td>EMAIL</td>
</tr>
<tr>
<td>EMP_EMP_ID_PK</td>
<td>EMPLOYEE_ID</td>
</tr>
<tr>
<td>EMP_HIRE_DATE_NN</td>
<td>HIRE_DATE</td>
</tr>
<tr>
<td>EMP_JOB_FK</td>
<td>JOB_ID</td>
</tr>
<tr>
<td>EMP_JOB_NN</td>
<td>JOB_ID</td>
</tr>
</tbody>
</table>

...
Summary

In this lesson, you should have learned how to create constraints.

• Types of constraints:
  – NOT NULL
  – UNIQUE
  – PRIMARY KEY
  – FOREIGN KEY
  – CHECK

• You can query the USER_CONSTRAINTS table to view all constraint definitions and names.
Practice 10 Overview

This practice covers the following topics:

• Adding constraints to existing tables
• Adding more columns to a table
• Displaying information in data dictionary views