

## CREATE TABLE: Example

```
CREATE TABLE customers
(
  customer_id          NUMBER(6)
, cust_first_name      VARCHAR2(20)
  CONSTRAINT cust_fname_nn NOT NULL
, cust_last_name       VARCHAR2(20)
  CONSTRAINT cust_lname_nn NOT NULL
, cust_address         cust_address_typ
, phone_numbers        phone_list_typ
, nls_language         VARCHAR2(3)
, nls_territory        VARCHAR2(30)
, credit_limit         NUMBER(9,2)
, cust_email           VARCHAR2(30)
, account_mgr_id       NUMBER(6)
, CONSTRAINT customer_credit_limit_max
  CHECK (credit_limit <= 5000)
, CONSTRAINT customer_id_min
  CHECK (customer_id > 0)
);
```

10-1

## CREATE TABLE: Example

The example in the slide shows the statement that is used to create the CUSTOMERS table in the OR schema.

## 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.

10-2

### Violating Constraints

When you have constraints in place on columns, an error is returned if you try to violate the constraint rule. For example, if you try to update a record with a value that is tied to an integrity constraint, an error is returned.

In the example in the slide, department 55 does not exist in the parent table, DEPARTMENTS, and so you receive the “parent key not found” violation ORA-02291.

## 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 (ORAL3HIST_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.
```

10-3

### Violating Constraints (continued)

If you attempt to delete a record with a value that is tied to an integrity constraint, an error is returned.

The example in the slide tries to delete department 60 from the `DEPARTMENTS` table, but it results in an error because that department number is used as a foreign key in the `EMPLOYEES` table. If the parent record that you attempt to delete has child records, you receive the “child record found” violation `ORA-02292`.

The following statement works because there are no employees in department 70:

```
DELETE FROM departments
WHERE department_id = 70;
```

```
1 rows deleted
```

## 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.

10-4

### Creating a Table Using a Subquery

A second method for creating a table is to apply the `AS subquery` clause, which both creates the table and inserts rows returned from the subquery.

In the syntax:

*table*

Is the name of the table

*column*

Is the name of the column, default value,

and integrity constraint

*subquery*

Is the `SELECT` statement that defines the

set of rows to be inserted into  
the new table

#### Guidelines

The table is created with the specified column names, and the rows retrieved by the `SELECT` statement are inserted into the table.

The column definition can contain only the column name and default value.

If column specifications are given, the number of columns must equal the number of columns in the subquery `SELECT` list.

If no column specifications are given, the column names of the table are the same as the column names in the subquery.

The column data type definitions and the `NOT NULL` constraint are passed to the new table. Note that only the explicit `NOT NULL` constraint will be inherited. The

PRIMARY KEY column will not pass the NOT NULL feature to the new column.  
Any other constraint rules are not passed to the new table. However, you can add constraints in the column definition.

## Creating a Table Using a Subquery

```
CREATE TABLE ord2458
AS
  SELECT order_id, order_date,
         order_status,
         customer_id
  FROM orders
  WHERE order_id = 2458;
```

CREATE TABLE succeeded.

```
DESCRIBE ord2458;
```

Name	Null	Type
ORDER_ID		NUMBER(12)
ORDER_DATE	NOT NULL	TIMESTAMP(6) WITH LOCAL TIME ZONE
ORDER_STATUS		NUMBER(2)
CUSTOMER_ID	NOT NULL	NUMBER(6)

10-5

### Creating a Table Using a Subquery (continued)

The example in the slide creates a table named `ORD2458`, which contains details of the order with ID 2458. Notice that the data for the `ORD2458` table comes from the `ORDERS` table.

You can verify the existence of a database table and check the column definitions by using the `DESCRIBE` command.

```
Error starting at line 1 in command:
CREATE TABLE dept80
  AS      SELECT employee_id, last_name,
                salary*12 ,
                hire_date0   FROM    employees   WHERE    department_id = 80
Error at Command Line:3 Column:18
```

#### Error report:

SQL Error: ORA-00998: must name this expression with a column alias  
00998. 00000 - "must name this expression with a column alias"

\*Cause:

\*Action: