

# Database Programming

PRIMARY KEY, FOREIGN KEY, and CHECK Constraints

# Objectives

This lesson covers the following objectives:

- Define and give an example of a PRIMARY KEY, FOREIGN KEY, and CHECK constraint
- Explain the purpose of defining PRIMARY KEY, FOREIGN KEY, and CHECK constraints
- Demonstrate the creation of constraints at the column level and table level in a CREATE TABLE statement
- Evaluate a business problem requiring the addition of a PRIMARY KEY and FOREIGN KEY constraint and write the code to execute the change

## Objectives (cont.)

This lesson covers the following objectives:

- Query the data dictionary for USER\_CONSTRAINTS and interpret the information returned

# Purpose

As discussed in the last section, constraints are used to prevent invalid data entry into database tables. What would happen if, surreptitiously or just through a careless mistake, your personal unique identification was given to another person? What if tomorrow at school someone else was credited with your classes for graduation or was able to eat lunch using your lunch-card number?

Ensuring data integrity is what constraints are all about. After all, you're unique!

# PRIMARY KEY Constraints

A PRIMARY KEY constraint is a rule that the values in one column or a combination of columns must uniquely identify each row in a table. No primary-key value can appear in more than one row in the table.

To satisfy a PRIMARY KEY constraint, both of the following conditions must be true:

- No column that is part of the primary key can contain a null.
- A table can have only one primary key.

## PRIMARY KEY Constraints (cont.)

PRIMARY KEY constraints can be defined at the column or the table level. However, if a composite PRIMARY KEY is created, it must be defined at the table level.

When defining PRIMARY KEY columns, it is a good practice to use the suffix `_pk` in the constraint name. For example, the constraint name for the PRIMARY KEY column named `id` in the `DJs on Demand d_events` table could be `d_events_id_pk`.

## PRIMARY KEY Constraints (cont.)

In a CREATE TABLE statement, the column-level PRIMARY KEY constraint syntax is stated:

```
CREATE TABLE clients
(client_number NUMBER(4) CONSTRAINT
client_client_num_pk PRIMARY KEY,
first_name VARCHAR2(14),
last_name VARCHAR2(13));
```

Note that the column-level simply refers to the area in the CREATE TABLE statement where the columns are defined. The table level refers to the last line in the statement below the list of individual column names.

## PRIMARY KEY Constraints (cont.)

To define a composite PRIMARY KEY, you must define the constraint at the table level rather than the column level. An example of a composite unique-key constraint name is:

```
CONSTRAINT id_venue_id_pk PRIMARY KEY (id, venue_id)
```



# FOREIGN KEY (REFERENTIAL INTEGRITY) Constraints

FOREIGN KEY constraints are also called "referential integrity" constraints.

```
CREATE TABLE clients
(client_number NUMBER(4) CONSTRAINT client_client_num_pk PRIMARY KEY,
 first_name VARCHAR2(14),
 last_name VARCHAR2(13),
 department_id VARCHAR2(4,0),
 CONSTRAINT clients_dept_id_fk FOREIGN KEY(department_id)
 REFERENCES departments(department_id));
```

Foreign Key constraints designate a column or combination of columns as a foreign key. A foreign key links back to the primary key (or a unique key) in another table, and this link is the basis of the relationship between tables.

## Stating a Foreign Key

To state a FOREIGN KEY constraint, use statements such as:

“The child table column named \_\_\_\_\_ with a data type of \_\_\_\_\_ has a CONSTRAINT named \_\_\_\_\_ which references its parent table called \_\_\_\_\_ which has a column called \_\_\_\_\_.”

## Stating a Foreign Key (cont.)

To state a table-level FOREIGN KEY constraint use statements such as:

“There is a table-level CONSTRAINT named \_\_\_\_\_ which is a FOREIGN KEY (in the \_\_\_\_\_ table); it REFERENCES the parent \_\_\_\_\_ table (which has a column named \_\_\_\_\_).”

# Viewing a Foreign Key

The table containing the foreign key is called the "child" table and the table containing the referenced key is called the "parent" table.

## D\_CLIENTS - Parents

CLIENT_NUMBER	FIRST_NAME	LAST_NAME	PHONE	EMAIL
5922	Hiram	Peters	3715832249	<a href="mailto:hpeters@yahoo.com">hpeters@yahoo.com</a>
5857	Serena	Jones	7035335900	<a href="mailto:serena.jones@jones.com">serena.jones@jones.com</a>
6133	Lauren	Vigil	4072220090	lbv@lbv.net

## D\_EVENTS - Child

ID	NAME	EVENT_DATE	DESCRIPTION	COST	VENUE_ID	PACKAGE_CODE	THEME_CODE	CLIENT_NUMBER
100	Peters Graduation	14-MAY-2004	Party for 200, red, white, blue motif	8000	100	112	200	5922
105	Vigil Wedding	28-APR-2004	Black tie at Four Seasons Hotel	10000	220	200	200	6133

## Viewing a Foreign Key (cont.)

In the tables shown, D\_CLIENTS primary-key client\_number also appears in D\_EVENTS as a foreign-key column.

### D\_CLIENTS - Parents

CLIENT_NUMBER	FIRST_NAME	LAST_NAME	PHONE	EMAIL
5922	Hiram	Peters	3715832249	<a href="mailto:hpeters@yahoo.com">hpeters@yahoo.com</a>
5857	Serena	Jones	7035335900	<a href="mailto:serena.jones@jones.com">serena.jones@jones.com</a>
6133	Lauren	Vigil	4072220090	lbv@lbv.net

### D\_EVENTS - Child

ID	NAME	EVENT_DATE	DESCRIPTION	COST	VENUE_ID	PACKAGE_CODE	THEME_CODE	CLIENT_NUMBER
100	Peters Graduation	14-MAY-2004	Party for 200, red, white, blue motif	8000	100	112	200	5922
105	Vigil Wedding	28-APR-2004	Black tie at Four Seasons Hotel	10000	220	200	200	6133

# Referential-integrity Constraint

To satisfy a referential-integrity constraint, a foreign-key value must match an existing value in the parent table or be NULL.

**D\_CLIENTS - Parents**

CLIENT_NUMBER	FIRST_NAME	LAST_NAME	PHONE	EMAIL
5922	Hiram	Peters	3715832249	<a href="mailto:hpeters@yahoo.com">hpeters@yahoo.com</a>
5857	Serena	Jones	7035335900	<a href="mailto:serena.jones@jones.com">serena.jones@jones.com</a>
6133	Lauren	Vigil	4072220090	lbv@lbv.net

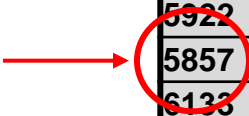
**D\_EVENTS - Child**

ID	NAME	EVENT_DATE	DESCRIPTION	COST	VENUE_ID	PACKAGE_CODE	THEME_CODE	CLIENT_NUMBER
100	Peters Graduation	14-MAY-2004	Party for 200, red, white, blue motif	8000	100	112	200	5922
105	Vigil Wedding	28-APR-2004	Black tie at Four Seasons Hotel	10000	220	200	200	6133

## Referential-integrity Constraint (cont.)

In the example, note that a primary-key value can exist without a corresponding foreign-key value; however, a foreign-key must have a corresponding primary key.

D\_CLIENTS - Parents



CLIENT_NUMBER	FIRST_NAME	LAST_NAME	PHONE	EMAIL
5922	Hiram	Peters	3715832249	<a href="mailto:hpeters@yahoo.com">hpeters@yahoo.com</a>
5857	Serena	Jones	7035335900	<a href="mailto:serena.jones@jones.com">serena.jones@jones.com</a>
6133	Lauren	Vigil	4072220090	lbv@lbv.net

D\_EVENTS - Child

ID	NAME	EVENT_DATE	DESCRIPTION	COST	VENUE_ID	PACKAGE_CODE	THEME_CODE	CLIENT_NUMBER
100	Peters Graduation	14-MAY-2004	Party for 200, red, white, blue motif	8000	100	112	200	5922
105	Vigil Wedding	28-APR-2004	Black tie at Four Seasons Hotel	10000	220	200	200	6133

# Referential-Integrity Constraint Rule

The rule is: before you define a referential-integrity constraint in the child table, the referenced UNIQUE or PRIMARY KEY constraint on the parent table must already be defined.

## D\_CLIENTS - Parents

CLIENT_NUMBER	FIRST_NAME	LAST_NAME	PHONE	EMAIL
5922	Hiram	Peters	3715832249	<a href="mailto:hpeters@yahoo.com">hpeters@yahoo.com</a>
5857	Serena	Jones	7035335900	<a href="mailto:serena.jones@jones.com">serena.jones@jones.com</a>
6133	Lauren	Vigil	4072220090	lbv@lbv.net

## D\_EVENTS - Child

ID	NAME	EVENT_DATE	DESCRIPTION	COST	VENUE_ID	PACKAGE_CODE	THEME_CODE	CLIENT_NUMBER
100	Peters Graduation	14-MAY-2004	Party for 200, red, white, blue motif	8000	100	112	200	5922
105	Vigil Wedding	28-APR-2004	Black tie at Four Seasons Hotel	10000	220	200	200	6133



## Referential-Integrity Constraint Rule (cont.)

In other words, you must first have a parent primary key defined before you can create a foreign key in a child table.

### D\_CLIENTS - Parents

CLIENT_NUMBER	FIRST_NAME	LAST_NAME	PHONE	EMAIL
5922	Hiram	Peters	3715832249	<a href="mailto:hpeters@yahoo.com">hpeters@yahoo.com</a>
5857	Serena	Jones	7035335900	<a href="mailto:serena.jones@jones.com">serena.jones@jones.com</a>
6133	Lauren	Vigil	4072220090	lbv@lbv.net

### D\_EVENTS - Child

ID	NAME	EVENT_DATE	DESCRIPTION	COST	VENUE_ID	PACKAGE_CODE	THEME_CODE	CLIENT_NUMBER
100	Peters Graduation	14-MAY-2004	Party for 200, red, white, blue motif	8000	100	112	200	5922
105	Vigil Wedding	28-APR-2004	Black tie at Four Seasons Hotel	10000	220	200	200	6133

# FOREIGN KEY Constraint

To define a FOREIGN KEY constraint, it is good practice to use the suffix `_fk` in the constraint name.

For example, the constraint name for the FOREIGN KEY column `song_id` in the `DJs on Demand` table named `d_track_listings` could be named `d_track_list_song_id_fk`.

# FOREIGN KEY Constraint Syntax

The syntax for defining a FOREIGN KEY constraint requires a reference to the table and column in the parent table. A FOREIGN KEY constraint in a CREATE TABLE statement can be defined as follows.

## Column-level syntax:

```
song_id NUMBER(5) CONSTRAINT d_track_list_ song_id_fk REFERENCES  
d_songs(id)
```

## Table-level syntax:

```
CONSTRAINT d_track_list_ song_id_fk FOREIGN KEY (song_id)  
REFERENCES d_songs(id)
```

## ON DELETE CASCADE - Maintaining Referential Integrity

Using the ON DELETE CASCADE option when defining a foreign key enables the dependent rows in the child table to be deleted when a row in the parent table is deleted.

If the foreign key does not have an ON DELETE CASCADE option, referenced rows in the parent table cannot be deleted. In other words, the child table FOREIGN KEY constraint includes the ON DELETE CASCADE permission allowing its parent to delete the rows that it refers to.

# ON DELETE CASCADE - Maintaining Referential Integrity (cont.)

## D\_CLIENTS - Parents

CLIENT_NUMBER	FIRST_NAME	LAST_NAME	PHONE	EMAIL
5922	Hiram	Peters	3715832249	<a href="mailto:hpeters@yahoo.com">hpeters@yahoo.com</a>
5857	Serena	Jones	7035335900	<a href="mailto:serena.jones@jones.com">serena.jones@jones.com</a>
6133	Lauren	Vigil	4072220090	lbv@lbv.net

## D\_EVENTS - Child

ID	NAME	EVENT_DATE	DESCRIPTION	COST	VENUE_ID	PACKAGE_CODE	THEME_CODE	CLIENT_NUMBER
100	Peters Graduation	14-MAY-2004	Party for 200, red, white, blue motif	8000	100	112	200	5922
105	Vigil Wedding	28-APR-2004	Black tie at Four Seasons Hotel	10000	220	200	200	6133

## ON DELETE CASCADE

If the `song_id` column in `D_TRACK_LISTINGS` was created with the `ON DELETE CASCADE` option specified, the `DELETE` statement issued on the `D_SONGS` table will execute. If the `ON DELETE CASCADE` option was not specified when the `song_id` column in `D_TRACK_LISTINGS` was created, the attempt to delete `song_id = 47` will fail.

**DELETE from D\_SONGS  
WHERE song\_id = 47**



**D\_TRACK\_LISTINGS**

SONG_ID	CD_NUMBER	TRACK
45	92	1
46	93	1
47	91	2
48	95	5
49	91	3
50	93	4

# ON DELETE CASCADE Syntax

## Column-level ON DELETE CASCADE syntax:

```
song_id NUMBER(5) CONSTRAINT  
d_track_list_ song_id_fk REFERENCES  
d_songs(id) ON DELETE CASCADE
```

## Table-level ON DELETE CASCADE syntax:

```
CONSTRAINT d_track_list_ song_id_fk  
FOREIGN KEY (song_id) REFERENCES  
d_songs(id) ON DELETE CASCADE
```

**DELETE from D\_SONGS  
WHERE song\_id = 47**



**D\_TRACK\_LISTINGS**

SONG_ID	CD_NUMBER	TRACK
45	92	1
46	93	1
<b>47</b>	<b>91</b>	<b>2</b>
48	95	5
49	91	3
50	93	4

## ON DELETE SET NULL

Rather than having the rows in the child table deleted when using an ON DELETE CASCADE option, the child rows can be filled with null values using the ON DELETE SET NULL option.

When do you choose whether to delete a row or simply set the values to null?



## ON DELETE SET NULL (cont.)

An example might be when the parent table value is being changed to a new number such as converting inventory numbers to bar-code numbers. You would not want to delete the rows in the child table.

When the new bar-code numbers are entered into the parent table, they would then be able to be inserted into the child table without having to totally re-create each child table row.

# CHECK Constraints

The CHECK constraint explicitly defines a condition that must be met. To satisfy the constraint, each row in the table must make the condition either True or unknown (due to a null). The condition of a CHECK constraint can refer to any column in the specified table, but not to columns of other tables.

# CHECK Constraint Example

What is each constraint limiting? The cd\_numbers must be between 10 and 999; year must be greater than 1996; the producer must be in the list shown.

```
CREATE d_cds (  
    cd_number NUMBER CONSTRAINT d_cds_cd_num_range  
        CHECK (cd_number BETWEEN 10 AND 999) ,  
    year NUMBER(4)  CONSTRAINT d_cds_year_min  
        CHECK (year > 1996) ,  
    producer VARCHAR2(10)  CONSTRAINT d_cds_prod_list  
        CHECK (producer IN ('Old Town Records', 'The Music Man',  
            'Middle Earth Records', 'R&B Inc', 'Tunes Are US')) ;
```

# CHECK Constraint Conditions

- A CHECK constraint must only be on the row where the constraint is defined.
- A CHECK constraint cannot be used in queries that refer to values in other rows.
- The CHECK constraint cannot contain calls to the functions SYSDATE, UID, USER, or USERENV. The statement `CHECK(SYSDATE > '05-MAY-1999')` is not allowed
- The CHECK constraint cannot use the pseudocolumns CURRVAL, NEXTVAL, LEVEL, or ROWNUM. The statement `CHECK(NEXTVAL > 0)` is not allowed.

## CHECK Constraint Conditions (cont.)

- A single column can have multiple CHECK constraints that reference the column in its definition. There is no limit to the number of CHECK constraints that you can define on a column.

# CHECK Constraint Syntax

CHECK constraints can be defined at the column level or the table level. The syntax to define a CHECK constraint is:

Column-level syntax:

```
salary NUMBER(8,2) CONSTRAINT f_staffs_min_salary CHECK (salary > 0)
```

Table-level syntax:

```
CONSTRAINT f_staffs_min_salary CHECK (salary > 0)
```

# Terminology

Key terms used in this lesson included:

- CHECK constraint
- FOREIGN KEY constraint
- NOT NULL
- ON DELETE CASCADE
- ON DELETE SET NULL
- PRIMARY KEY constraint

# Summary

In this lesson, you should have learned how to:

- Define and give an example of a PRIMARY KEY, FOREIGN KEY, and CHECK constraint
- Explain the purpose of defining PRIMARY KEY, FOREIGN KEY, and CHECK constraints
- Demonstrate the creation of constraints at the column level and table level in a CREATE TABLE statement
- Evaluate a business problem requiring the addition of a PRIMARY KEY and FOREIGN KEY constraint and write the code to execute the change



## Summary (cont.)

In this lesson, you should have learned how to:

- Query the data dictionary for USER\_CONSTRAINTS and interpret the information returned