Database Design and SQL

Chapter 7: Database Constraints

FSDM/CPCM

Sagara Samarawickrama

2023S

Chapter Objectives

- Explain the purpose of constraints
- Use the primary key constraints
- Use unique constraints.
- Use foreign key constraints
- Use check constraints
- Create constraints using create CREATE TABLE and ALTER TABLE command
- Use the ALTER TABLE command to add and drop constraints for an existing table

Database Constraints

Introduction to Constraints:

A constraint sometimes referred as integrity constraint or integrity rule, is a restriction based on a business rule or other business policy that ensure data in the database acceptable and accurate. For example , a primary key constraint defines a unique identification of each row in the table.

Each DBMS provides features that are used to specify integrity constraints. The features usually include range checks(e.g. “between 1 to 100”) or allowable values(“ equal to AB,ON or MB).More complex constrains may be specified as well; examples include relationships between columns within the same table(e.g employees hire date must be greater than his birthdate).The dbms which handles all updating of the database , checks these constraints when rows are inserted ,updated or deleted. If a constraint is not met , the management system blocks the operation and signals an error.

Database Constraints

Three integrity rule categories allow the specification of important constraints that the management system automatically enforces whenever a database update occurs;

- Data Integrity
- Entity Integrity
- Referential Integrity

These rules protect not only the specified values in columns but also the identity and interrelationships of rows

Database Constraints

Data integrity

Data integrity defines the possible values of a column. In a database system, data integrity is defined by

- Data type and length
- Null value acceptance
- Allowable values
- Default value

For example , if an age column in an EMPLOYEES table is defined as an integer, the value of every instance of that column must always be numeric and an integer.

Database Constraints

Entity Integrity

This is fairly straightforward concept: Every row in a table represents an entity that must exist in the real world; therefore ,every row must be uniquely identifiable. It follows that no completely duplicate rows can exist in a table; otherwise , the unique existence of entities is not represented in the database.

The primary key is a set of columns whose values are unique for all rows in the table. Because of this fact, the primary key forms the only means of addressing a specific row in the relational database model.

A consequence of the requirement for unique primary key values is that none of the values in a rows primary key columns can be null; that none can be missing or unknown.

Database Constraints

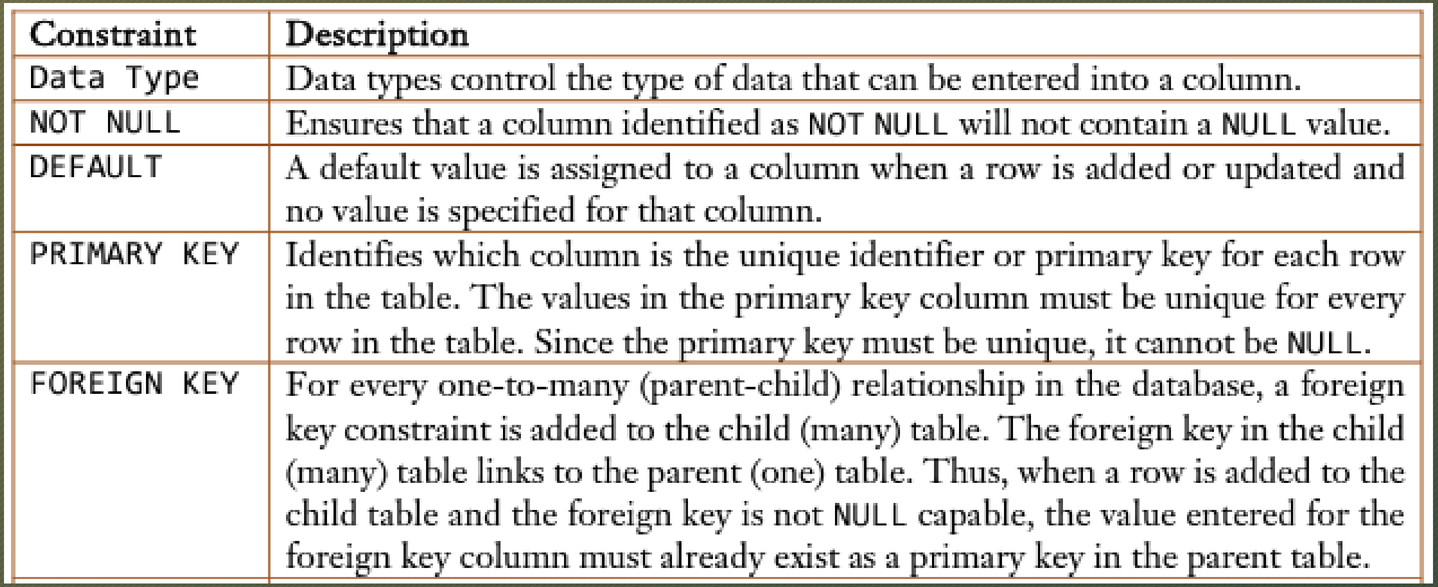
Referential Integrity

Referential integrity is a database concept that ensures that relationships between tables remain reliable. That is, when a table contains a foreign key associated with another tables primary key, the concept of referential integrity states that a row contains the foreign key may not be added to the table unless a matching value exists in the primary key column of the parent table. In addition , referential integrity includes the concepts of cascading updates and deletes, which ensure that changes made to the table with the foreign key, are reflected in the primary table

Database Constraints

Constraint Types:

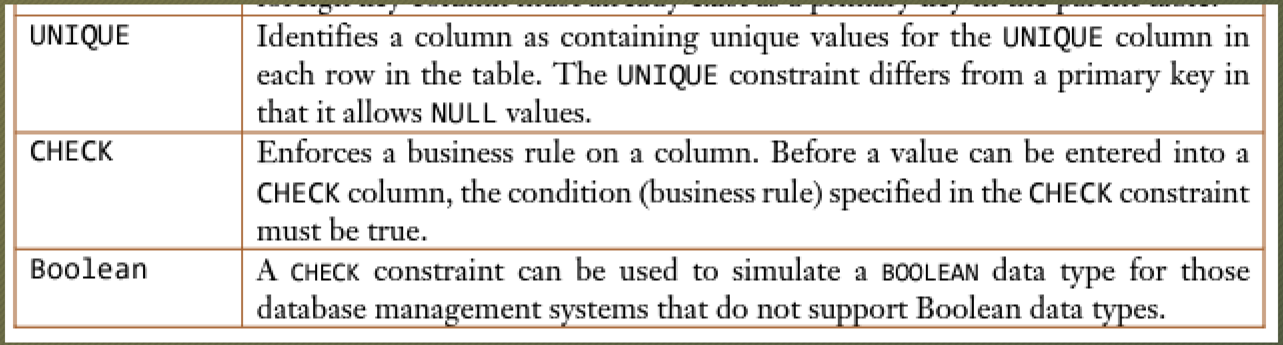
Database constraints provide a way to guarantee that rows in a table have valid primary or unique key values that, that rows in a dependent table a valid foreign key values that reference rows in a parent table, and that individual column values are valid.This chapter discusses five constraints , identified in Figure shown below



Constraint	Description
Data Type	Data types control the type of data that can be entered into a column.
NOT NULL	Ensures that a column identified as NOT NULL will not contain a NULL value.
DEFAULT	A default value is assigned to a column when a row is added or updated and no value is specified for that column.
PRIMARY KEY	Identifies which column is the unique identifier or primary key for each row in the table. The values in the primary key column must be unique for every row in the table. Since the primary key must be unique, it cannot be NULL.
FOREIGN KEY	For every one-to-many (parent-child) relationship in the database, a foreign key constraint is added to the child (many) table. The foreign key in the child (many) table links to the parent (one) table. Thus, when a row is added to the child table and the foreign key is not NULL capable, the value entered for the foreign key column must already exist as a primary key in the parent table.

Database Constraints

Constraint Types:



UNIQUE	Identifies a column as containing unique values for the UNIQUE column in each row in the table. The UNIQUE constraint differs from a primary key in that it allows NULL values.
CHECK	Enforces a business rule on a column. Before a value can be entered into a CHECK column, the condition (business rule) specified in the CHECK constraint must be true.
Boolean	A CHECK constraint can be used to simulate a BOOLEAN data type for those database management systems that do not support Boolean data types.

Data Type Constraint

When a data type is specified for a column, the management system restricts the contents of that column to that data type. For example, if a column is defined as an INTEGER data type, only whole numbers are allowed to be entered into the column.

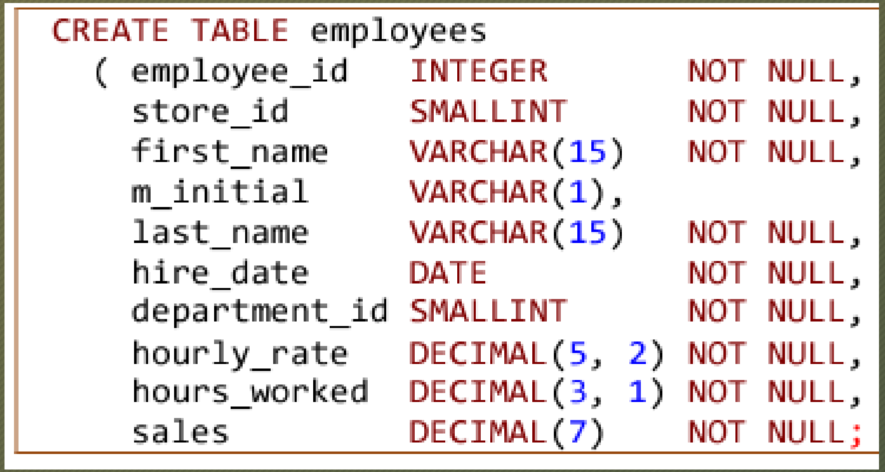
Database Constraints

NOT NULL Constraint:

Occasionally, the actual data value for a column is unknown when a row is inserted into a table or when an existing row is modified. A special value called NULL is used to indicate that the value for the column is unknown. In other words, if a column contains a NULL values, the actual value of the column is unknown.

Database Constraints

The NOT NULL clause can be used in the CREATE TABLE command to indicate that a column cannot contain null values. If a column does not allow the null value, a value must be assigned to the column-either a default value or a user-supplied value.



```
CREATE TABLE employees
  ( employee_id   INTEGER       NOT NULL,
    store_id      SMALLINT      NOT NULL,
    first_name    VARCHAR(15)   NOT NULL,
    m_initial     VARCHAR(1),
    last_name     VARCHAR(15)   NOT NULL,
    hire_date     DATE          NOT NULL,
    department_id SMALLINT      NOT NULL,
    hourly_rate   DECIMAL(5, 2) NOT NULL,
    hours_worked  DECIMAL(3, 1) NOT NULL,
    sales         DECIMAL(7)    NOT NULL;
```

Database Constraints

Default Value Constraint:

A default value is assigned to a column when a row is added or updated and no value is specified for that column. Remember , NULL is a value .If a specific default value is not defined for a column, the system default value is used.

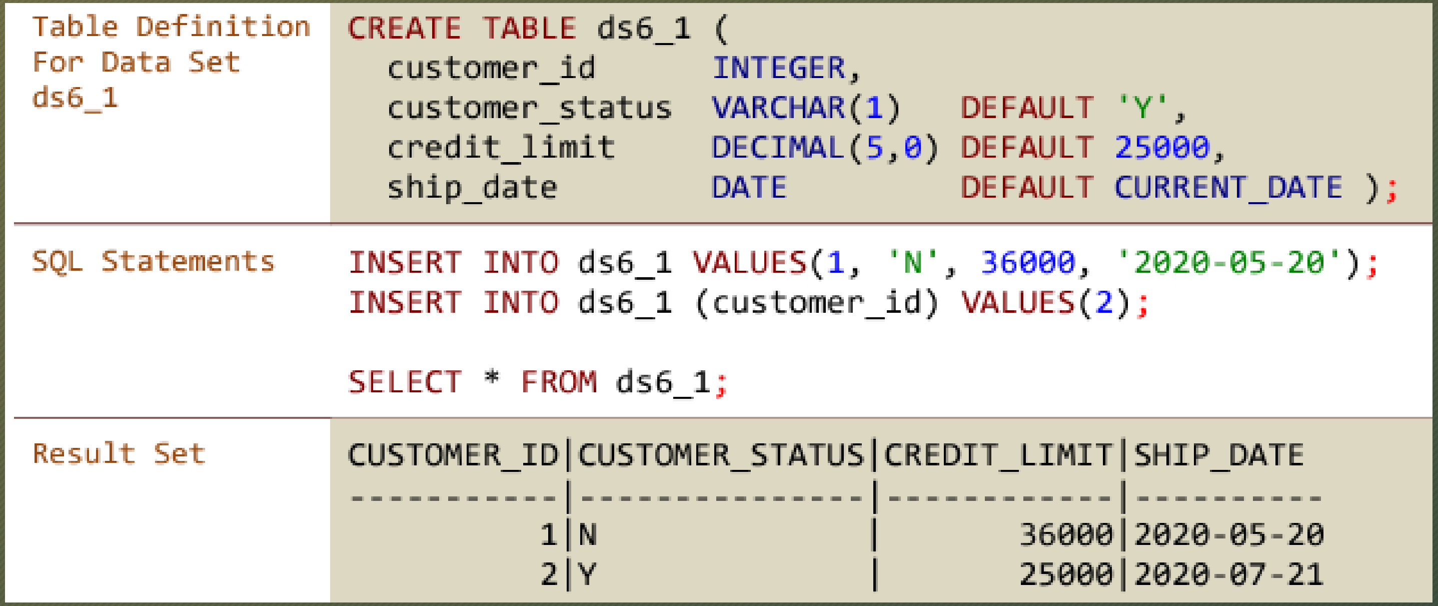


Table Definition For Data Set ds6_1

```
CREATE TABLE ds6_1 (
  customer_id       INTEGER,
  customer_status   VARCHAR(1)   DEFAULT 'Y',
  credit_limit      DECIMAL(5,0) DEFAULT 25000,
  ship_date         DATE         DEFAULT CURRENT_DATE );
```

SQL Statements

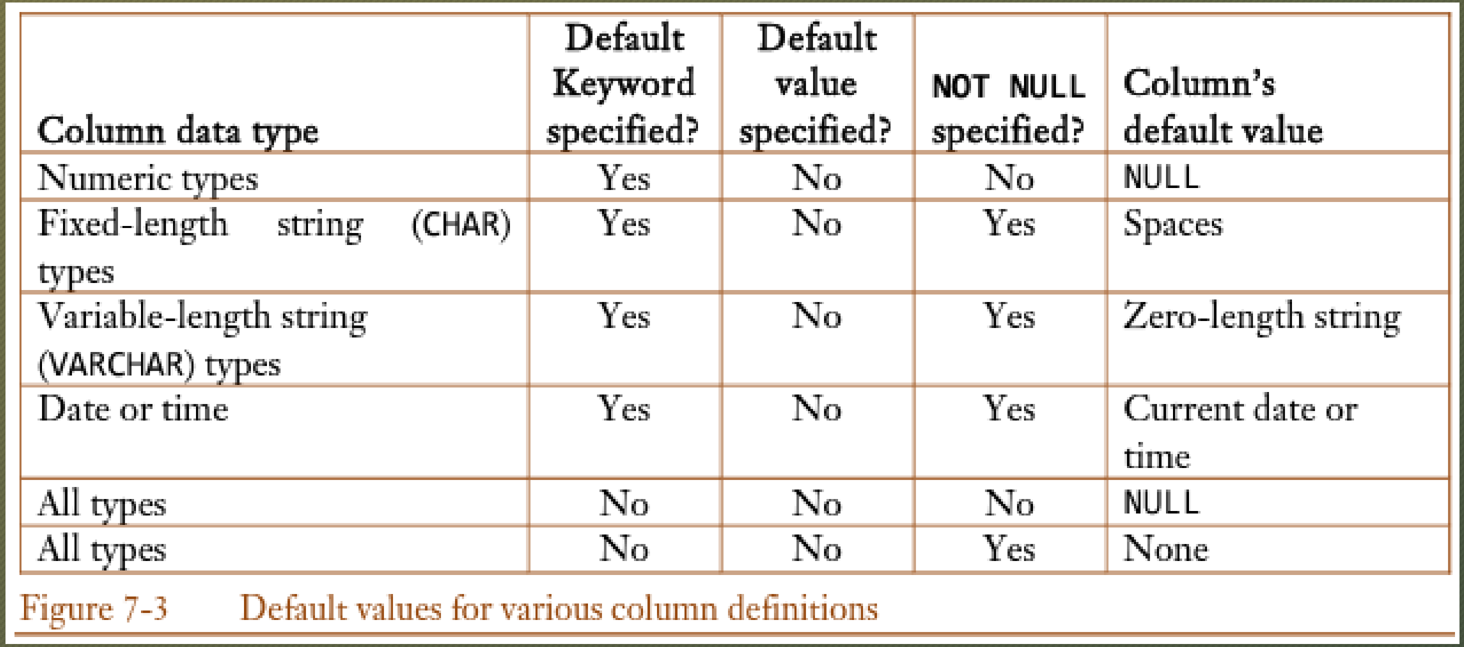
```
INSERT INTO ds6_1 VALUES(1, 'N', 36000, '2020-05-20');
INSERT INTO ds6_1 (customer_id) VALUES(2);

SELECT * FROM ds6_1;
```

Result Set

```
CUSTOMER_ID|CUSTOMER_STATUS|CREDIT_LIMIT|SHIP_DATE
-----------|---------------|------------|----------
          1|N              |       36000|2020-05-20
          2|Y              |       25000|2020-07-21
```

Database Constraints

Default Value Constraint:



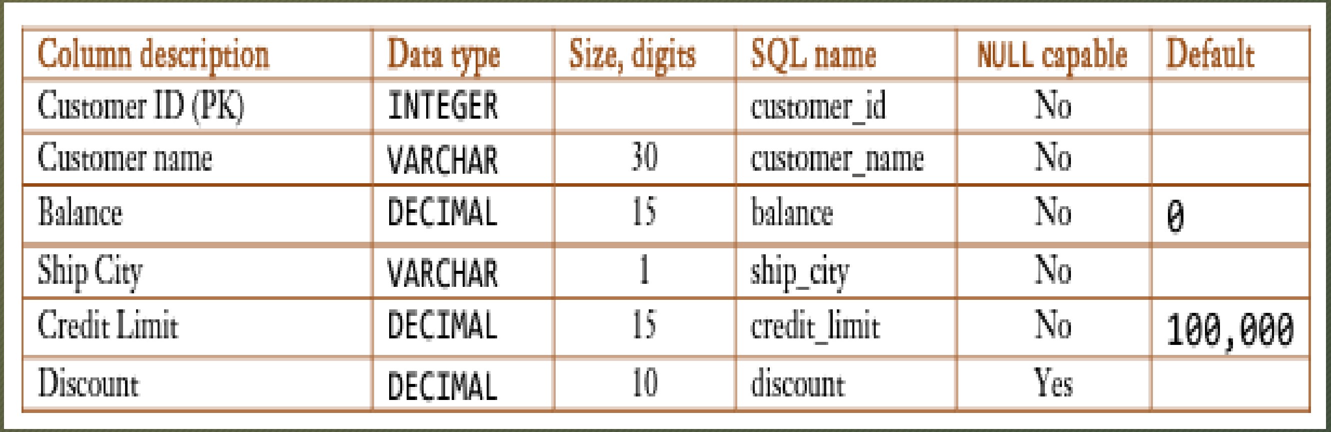
Column data type	Default Keyword specified?	Default value specified?	NOT NULL specified?	Column's default value
Numeric types	Yes	No	No	NULL
Fixed-length string (CHAR) types	Yes	No	Yes	Spaces
Variable-length string (VARCHAR) types	Yes	No	Yes	Zero-length string
Date or time	Yes	No	Yes	Current date or time
All types	No	No	No	NULL
All types	No	No	Yes	None

Figure 7-3 Default values for various column definitions

Database Constraints

Primary Key Constraint:

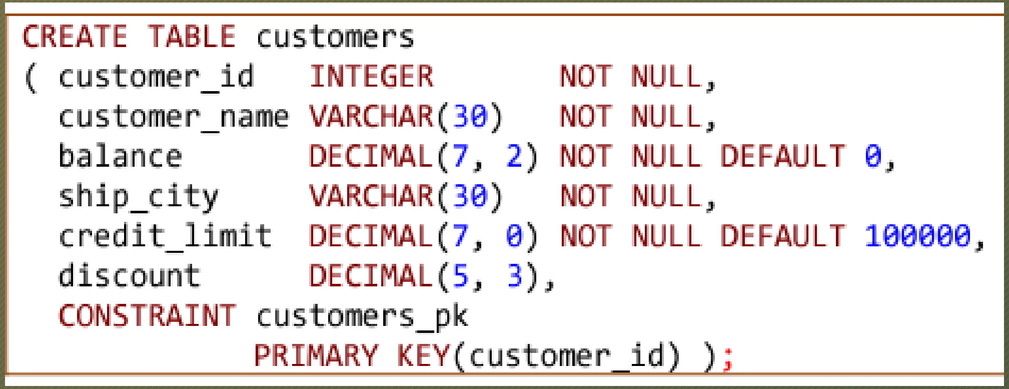
Each row in a table contains one or more columns that uniquely identify that row in the table. This single column or group of columns is referred to as a primary key and serves as the unique identifier for each row in the table. Primary key constraint can be added at the table level as part of the CREATE TABLE command



Column description	Data type	Size, digits	SQL name	NULL capable	Default
Customer ID (PK)	INTEGER		customer_id	No	
Customer name	VARCHAR	30	customer_name	No	
Balance	DECIMAL	15	balance	No	0
Ship City	VARCHAR	1	ship_city	No	
Credit Limit	DECIMAL	15	credit_limit	No	100,000
Discount	DECIMAL	10	discount	Yes	

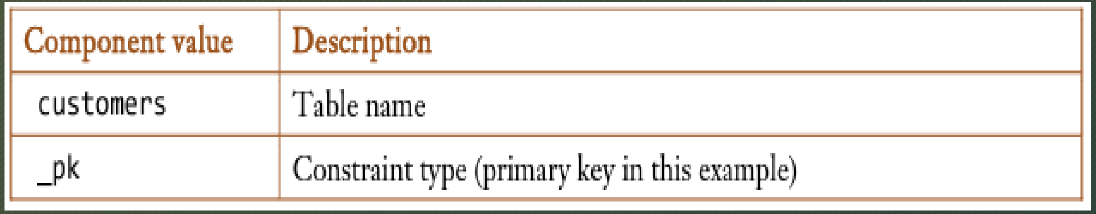
Database Constraints

Primary Key Constraint:



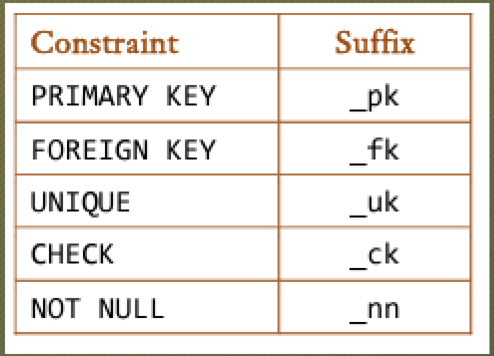
```
CREATE TABLE customers
( customer_id   INTEGER       NOT NULL,
  customer_name VARCHAR(30)   NOT NULL,
  balance       DECIMAL(7, 2) NOT NULL DEFAULT 0,
  ship_city     VARCHAR(30)   NOT NULL,
  credit_limit  DECIMAL(7, 0) NOT NULL DEFAULT 100000,
  discount      DECIMAL(5, 3),
  CONSTRAINT customers_pk
             PRIMARY KEY(customer_id) );
```

When constraint is defined, each constraint clause can optionally begin with the CONSTRAINT keyword followed by an optional constraint name. If a constraint name is not specified , the management system generates a name automatically



Component value	Description
customers	Table name
_pk	Constraint type (primary key in this example)

Database Constraints



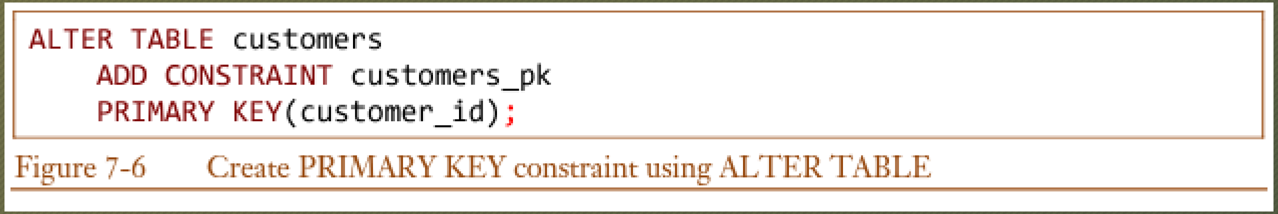
Constraint	Suffix
PRIMARY KEY	_pk
FOREIGN KEY	_fk
UNIQUE	_uk
CHECK	_ck
NOT NULL	_nn

Every primary key constraint must include NOT NULL. The system blocks any attempt to insert or update a row that would cause two rows in the same table to have identical value(s) for their primary key column(s).A table definition can have no more than one primary key constraint.

Database Constraints

Using the Alter Table Command to Add a Constraint

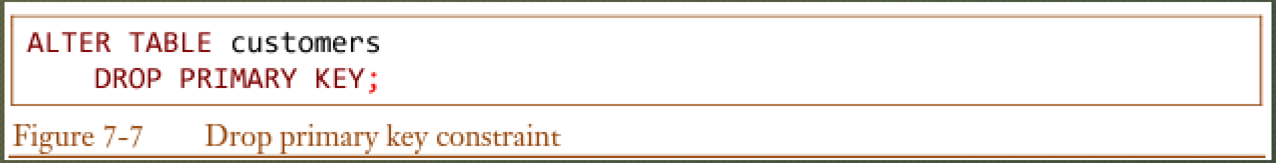
Primary key constraint can be added to a table with the ALTER TABLE command.



```
ALTER TABLE customers
    ADD CONSTRAINT customers_pk
    PRIMARY KEY(customer_id);
```

Figure 7-6 Create PRIMARY KEY constraint using ALTER TABLE

A constraint name is not required to drop a primary constraint with the table ALTER Command



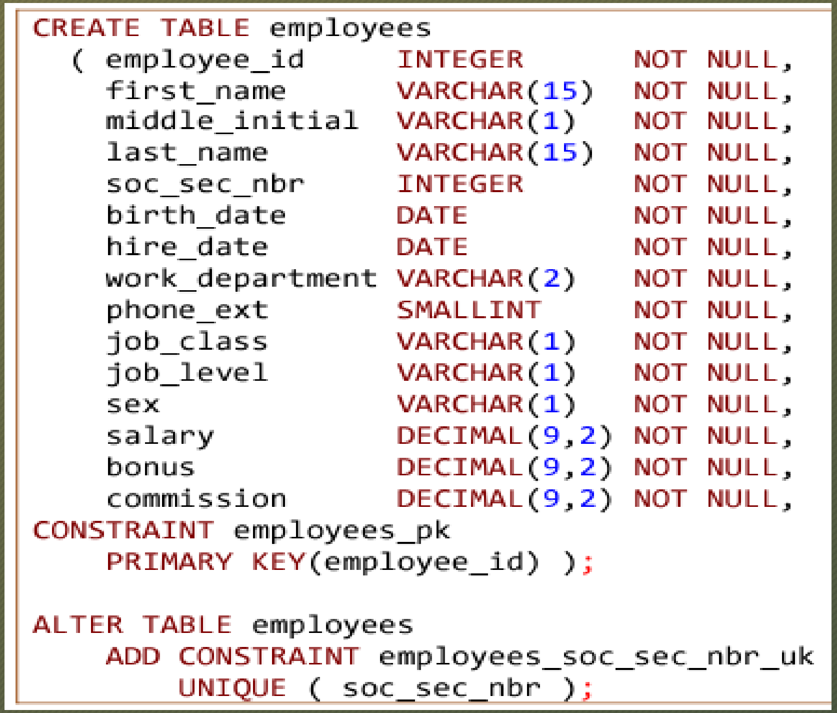
```
ALTER TABLE customers
    DROP PRIMARY KEY;
```

Figure 7-7 Drop primary key constraint

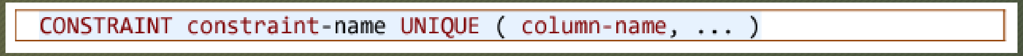
Database Constraints

Unique Constraint:

Unique key constraint is similar to a primary key constraint; however, a column listed as a unique constraint does not need to be defined with NOT NULL. It is recommended that a constraint name be specified for unique constraint:



```
CREATE TABLE employees
  ( employee_id      INTEGER      NOT NULL,
    first_name       VARCHAR(15)  NOT NULL,
    middle_initial   VARCHAR(1)   NOT NULL,
    last_name        VARCHAR(15)  NOT NULL,
    soc_sec_nbr      INTEGER      NOT NULL,
    birth_date       DATE         NOT NULL,
    hire_date        DATE         NOT NULL,
    work_department  VARCHAR(2)   NOT NULL,
    phone_ext        SMALLINT     NOT NULL,
    job_class        VARCHAR(1)   NOT NULL,
    job_level        VARCHAR(1)   NOT NULL,
    sex              VARCHAR(1)   NOT NULL,
    salary           DECIMAL(9,2) NOT NULL,
    bonus            DECIMAL(9,2) NOT NULL,
    commission       DECIMAL(9,2) NOT NULL,
CONSTRAINT employees_pk
    PRIMARY KEY(employee_id) );

ALTER TABLE employees
    ADD CONSTRAINT employees_soc_sec_nbr_uk
        UNIQUE ( soc_sec_nbr );
```



```
CONSTRAINT constraint-name UNIQUE ( column-name, ... )
```

Database Constraints

Foreign Key Constraint:

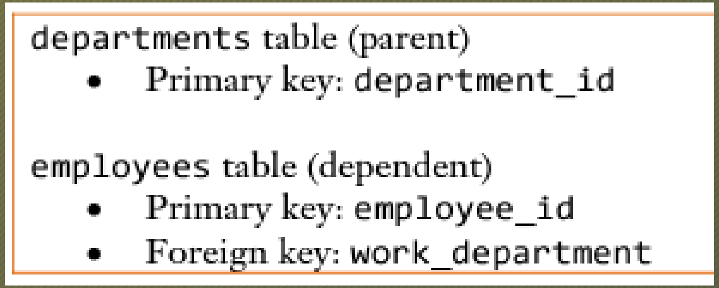
A foreign key is one or more columns in the child(dependent) table that contain values that match the primary key of a parent table. When a logical model is translated to a physical database, the relationships between tables are implemented as a foreign key constraints, sometimes referred to as a referential integrity.

A foreign key constraint specifies how rows in different tables are related and how the system handles row insert, delete, and update operations that might violate the relationship. For example order rows are generally related to the customers who place the orders. Although it might be valid for customer row to exist without any corresponding order of rows, it normally would be invalid for an order row not to have reference to a reference to a valid customer.

The table with the table with the primary key is called the parent table, and the table with the foreign key is called the dependent or child table

Database Constraints

Defining a Foreign Key Constraint:



departments table (parent)

- Primary key: department_id

employees table (dependent)

- Primary key: employee_id
- Foreign key: work_department

For each row in the employee table , the work department column must contain the same value as the department_id column of some row in the departments table because of this value tells which department the employee works in.The purpose of specifying a foreign key constraint is to ensure that the employee table never has a row with a non-null value in the work department column that has no matching row in the department table.

Database Constraints

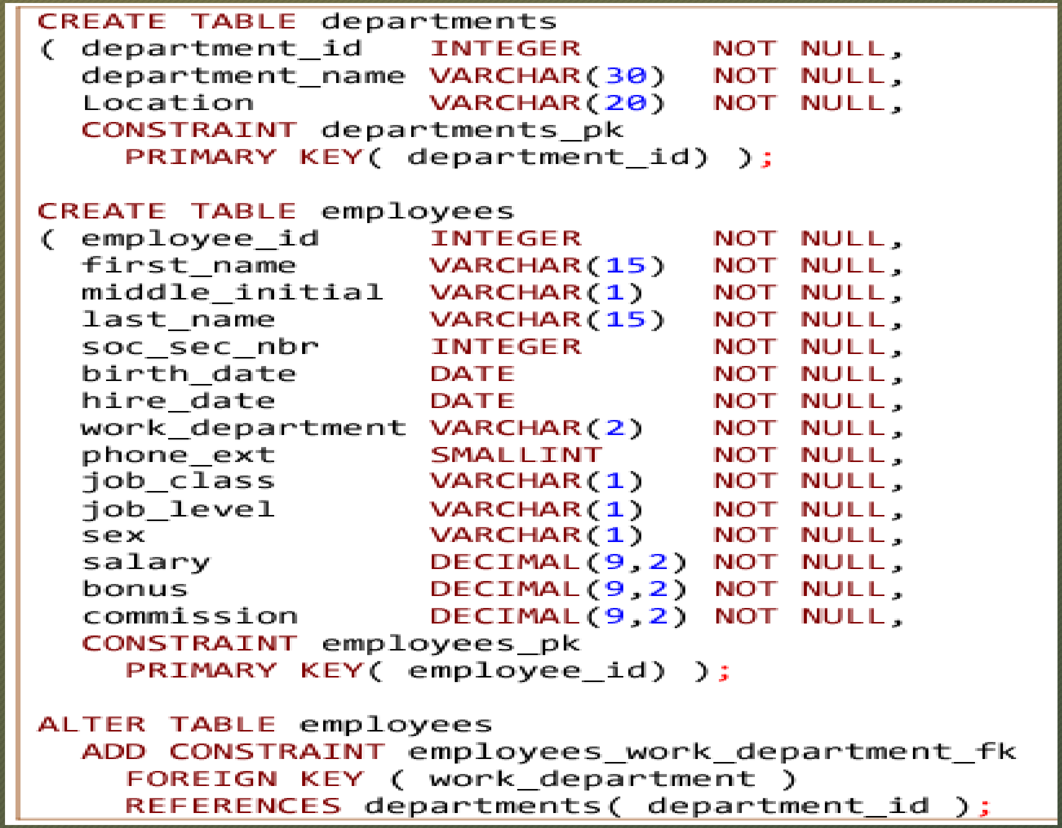
A primary key constraint should be defined for the parent table before the parent table foreign key constraints are defined for any dependent tables.

A foreign key constraint is specified in the dependent table. The constraint specifies the foreign key column(s) in the same table as the constraint and specifies the column(s) of the primary key or unique constraint in the parent table. The corresponding primary and foreign key columns must have identical data type length or precision.

Foreign key constraints are defined using the FOREIGN KEY clause, which consists of three components

- A constraint name
- The columns making up the foreign key
- A reference clause

Database Constraints



```
CREATE TABLE departments
( department_id    INTEGER       NOT NULL,
  department_name  VARCHAR(30)   NOT NULL,
  Location         VARCHAR(20)   NOT NULL,
  CONSTRAINT departments_pk
    PRIMARY KEY( department_id) );

CREATE TABLE employees
( employee_id      INTEGER       NOT NULL,
  first_name       VARCHAR(15)   NOT NULL,
  middle_initial   VARCHAR(1)    NOT NULL,
  last_name        VARCHAR(15)   NOT NULL,
  soc_sec_nbr      INTEGER       NOT NULL,
  birth_date       DATE          NOT NULL,
  hire_date        DATE          NOT NULL,
  work_department  VARCHAR(2)    NOT NULL,
  phone_ext        SMALLINT      NOT NULL,
  job_class        VARCHAR(1)    NOT NULL,
  job_level        VARCHAR(1)    NOT NULL,
  sex              VARCHAR(1)    NOT NULL,
  salary           DECIMAL(9,2)  NOT NULL,
  bonus            DECIMAL(9,2)  NOT NULL,
  commission       DECIMAL(9,2)  NOT NULL,
  CONSTRAINT employees_pk
    PRIMARY KEY( employee_id) );

ALTER TABLE employees
  ADD CONSTRAINT employees_work_department_fk
    FOREIGN KEY ( work_department )
    REFERENCES departments( department_id );
```

Database Constraints

The primary key of the deprtments table is department_id and the primary key of the employees table is employee_id

Lets examine the foreign key that relates the employee job table to the department table. This foreign key constraint ensures that no department code exist in the work_department column of the employee job table before the department code exists as a primary key in the department table. The work_department column in each row of the employee job table must contain a value of a department_id column in the department table. This foreign key ensures that

- A value in the work_department column of the employee job table cannot subsequently to a value that is not a valid department_id in the department table
- The primary key department_id of the department table cannot be deleted without the appropriate check for corresponding values in the employee job table foreign key column

Database Constraints

- To ensure that this integrity remains intact, a series of rules for inserting, deleting and updating applies.
- When inserting a row with a foreign key , the values of the foreign key columns are checked against the values of the primary key columns in the parent table. If no matching primary keys are found, the insert is disallowed
- A new primary key row can be inserted into parent table as long as the primary key is unique for the table
- When updating foreign key values, the same checks are performed as when a row with a foreign key
- Primary key values should never be changed . However , if the primary key vaue of or be set to NULL before the value of the primary key can be changed
- Deleting a row in the department table with a foreign key is always permitted

Database Constraints

- When deleting a row in the primary key has foreign key relationships with other tables, the action taken is indicated in the command; it restrict the deletion, cascades deletes to foreign key rows, or sets all referenced foreign keys to null

Foreign Key Actions

Foreign key constraints define the actions that will be taken on foreign key rows when a primary key is deleted. Four options can be specified

- Restrict : Disallows the deletion of the primary key row if any foreign keys relate to the row
- Cascade: Allows the deletion of the primary key row and also deletes the foreign key rows that relate to it
- Set Null: Allows the deletion of the primary key row and , instead of deleting all related foreign key rows, sets the foreign key columns to null

Database Constraints

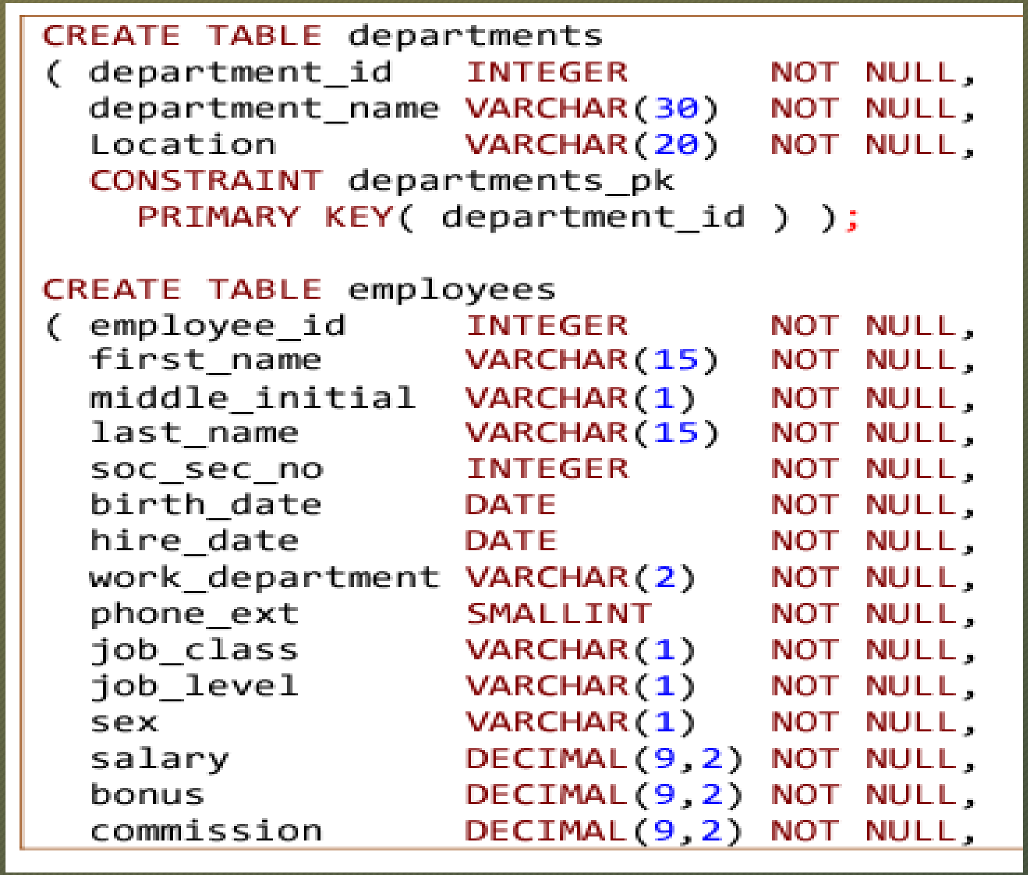
Check Constraint:

Check Constraints are used to enforce business rules by placing restrictions on the data that can be entered into a column. Any attempt to modify the column data (e.g. during INSERT or UPDATE processing) will cause the check constraints to be evaluated. If the modification conforms to the check condition , the modification is permitted to continue ; if not, the command will fail with a constraint violation.

Defining Check Constraints:

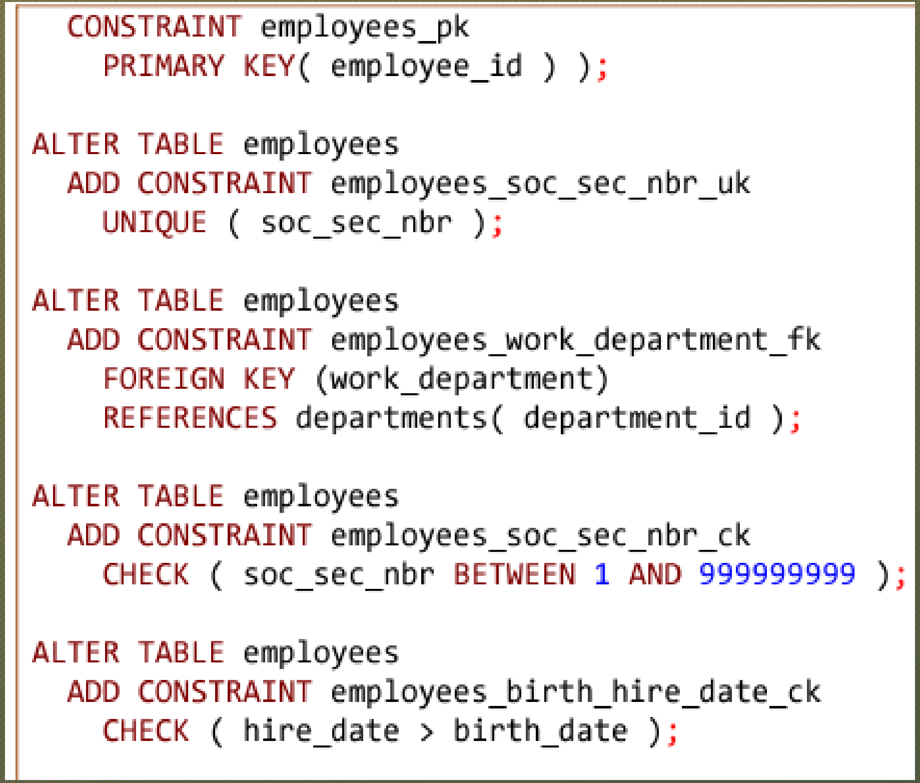
With every INSERT and UPDATE operation, the management system checks to ensure there are no check constraint violations before allowing the insert or update operation. If a check constraint condition evaluates true, the constraint is considered valid and the operation is performed. If the constraint is violated and a constraint violation error is returned

Database Constraints

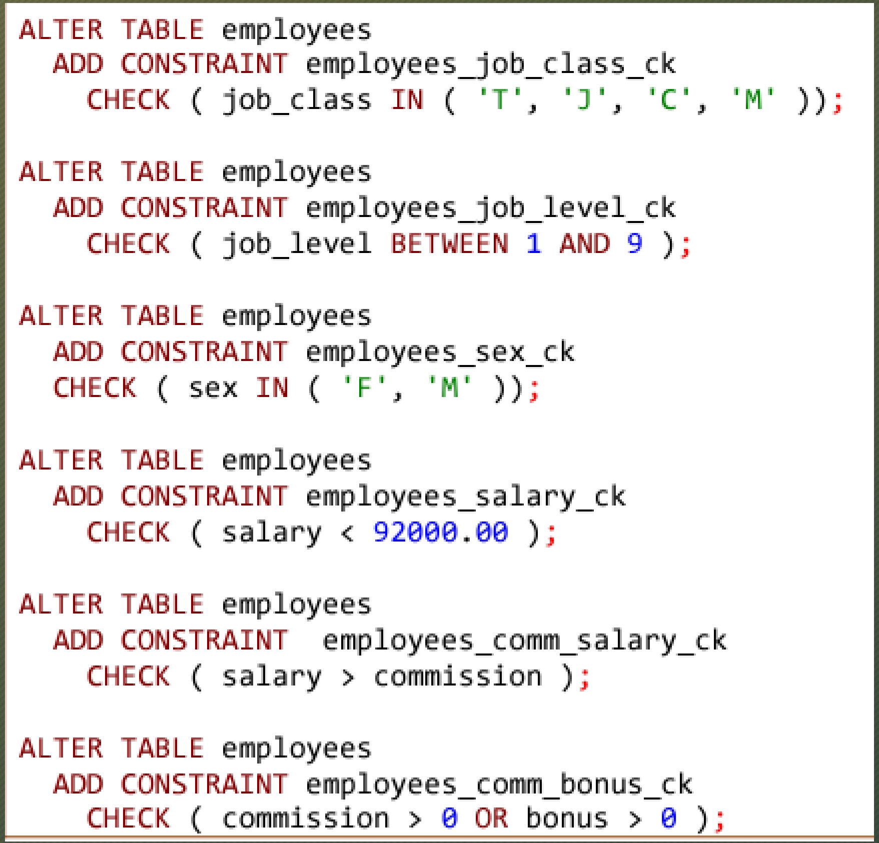


```
CREATE TABLE departments
( department_id    INTEGER       NOT NULL,
  department_name  VARCHAR(30)   NOT NULL,
  Location         VARCHAR(20)   NOT NULL,
  CONSTRAINT departments_pk
    PRIMARY KEY( department_id ) );

CREATE TABLE employees
( employee_id      INTEGER       NOT NULL,
  first_name       VARCHAR(15)   NOT NULL,
  middle_initial   VARCHAR(1)    NOT NULL,
  last_name        VARCHAR(15)   NOT NULL,
  soc_sec_no       INTEGER       NOT NULL,
  birth_date       DATE          NOT NULL,
  hire_date        DATE          NOT NULL,
  work_department  VARCHAR(2)    NOT NULL,
  phone_ext        SMALLINT      NOT NULL,
  job_class        VARCHAR(1)    NOT NULL,
  job_level        VARCHAR(1)    NOT NULL,
  sex              VARCHAR(1)    NOT NULL,
  salary           DECIMAL(9,2)  NOT NULL,
  bonus            DECIMAL(9,2)  NOT NULL,
  commission       DECIMAL(9,2)  NOT NULL,
```

Database Constraints



```sql
  CONSTRAINT employees_pk
    PRIMARY KEY( employee_id ) );

ALTER TABLE employees
  ADD CONSTRAINT employees_soc_sec_nbr_uk
    UNIQUE ( soc_sec_nbr );

ALTER TABLE employees
  ADD CONSTRAINT employees_work_department_fk
    FOREIGN KEY (work_department)
    REFERENCES departments( department_id );

ALTER TABLE employees
  ADD CONSTRAINT employees_soc_sec_nbr_ck
    CHECK ( soc_sec_nbr BETWEEN 1 AND 999999999 );

ALTER TABLE employees
  ADD CONSTRAINT employees_birth_hire_date_ck
    CHECK ( hire_date > birth_date );
```

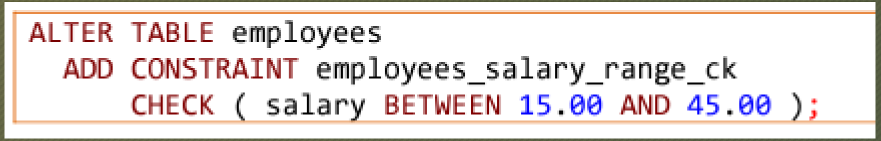


```sql
ALTER TABLE employees
  ADD CONSTRAINT employees_job_class_ck
    CHECK ( job_class IN ( 'T', 'J', 'C', 'M' ));

ALTER TABLE employees
  ADD CONSTRAINT employees_job_level_ck
    CHECK ( job_level BETWEEN 1 AND 9 );

ALTER TABLE employees
  ADD CONSTRAINT employees_sex_ck
  CHECK ( sex IN ( 'F', 'M' ));

ALTER TABLE employees
  ADD CONSTRAINT employees_salary_ck
    CHECK ( salary < 92000.00 );

ALTER TABLE employees
  ADD CONSTRAINT  employees_comm_salary_ck
    CHECK ( salary > commission );

ALTER TABLE employees
  ADD CONSTRAINT employees_comm_bonus_ck
    CHECK ( commission > 0 OR bonus > 0 );
```

Database Constraints

Range Check :

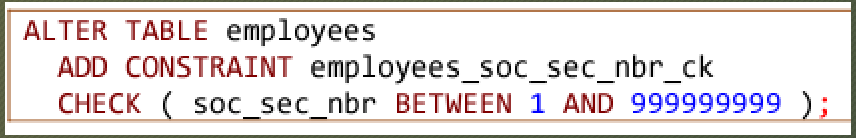
A range check constraint compares a column values to lower and upper limits that represent a range of values. The lower value is the lowest value that can be inserted into the column and the upper value is the highest values that can be inserted into the column



```
ALTER TABLE employees
  ADD CONSTRAINT employees_salary_range_ck
      CHECK ( salary BETWEEN 15.00 AND 45.00 );
```

In this check constraint , the salary is being verified that it contains a value between 15.00 and 45.00

Another example

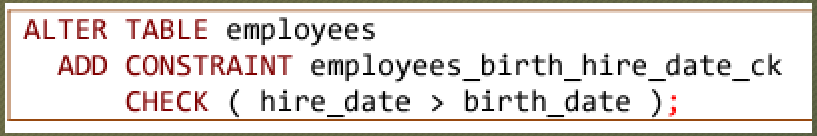


```
ALTER TABLE employees
  ADD CONSTRAINT employees_soc_sec_nbr_ck
  CHECK ( soc_sec_nbr BETWEEN 1 AND 999999999 );
```

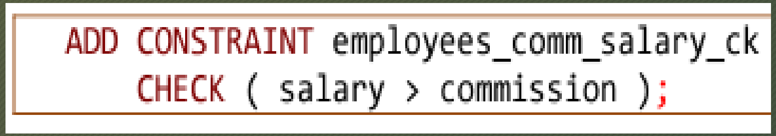
Database Constraints

Compare two columns:

The check constraint can compare two columns in the same table.For example the employees_birth_hire_date_ck check constraint compares two data columns to ensure that the employee's hire date (hire_date) is more recent than the birth date(birth_date).



```
ALTER TABLE employees
  ADD CONSTRAINT employees_birth_hire_date_ck
      CHECK ( hire_date > birth_date );
```

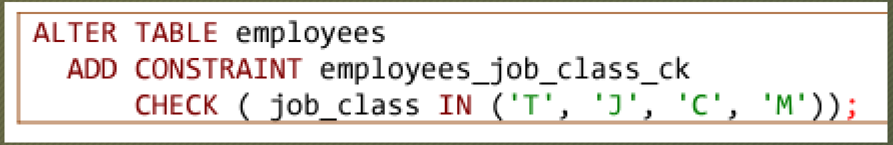


```
  ADD CONSTRAINT employees_comm_salary_ck
      CHECK ( salary > commission );
```

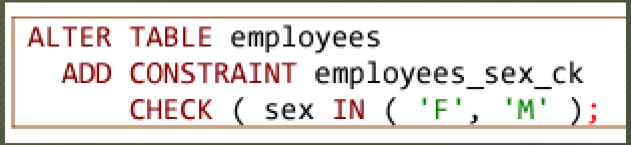
Database Constraints

List Of values:

The Job class column identifies the position of the employee within the company. Valid job classifications are T=Training, J=Junior , C=Clerk and M=Manager.



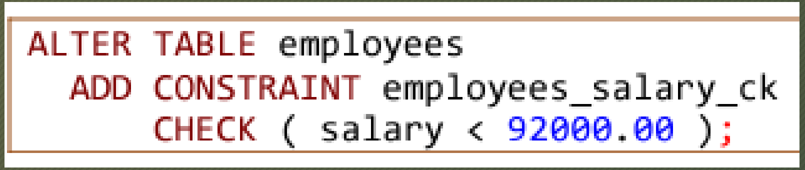
```
ALTER TABLE employees
  ADD CONSTRAINT employees_job_class_ck
      CHECK ( job_class IN ('T', 'J', 'C', 'M'));
```



```
ALTER TABLE employees
  ADD CONSTRAINT employees_sex_ck
      CHECK ( sex IN ( 'F', 'M' );
```

Limit Check :

At present no one at the company has salary greater that $92,000.00 .

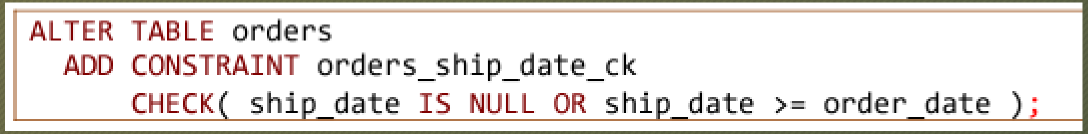


```
ALTER TABLE employees
  ADD CONSTRAINT employees_salary_ck
      CHECK ( salary < 92000.00 );
```

Database Constraints

NULL :

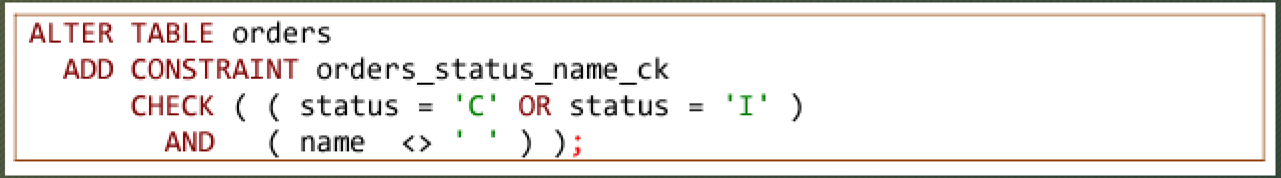
Check constraints can evaluated to true, false , or unknown. One way that a condition can evaluate to unknown is when the column is null.



```
ALTER TABLE orders
  ADD CONSTRAINT orders_ship_date_ck
      CHECK( ship_date IS NULL OR ship_date >= order_date );
```

To prevent a null value in a column and avoid the possibility of an unknown condition, simply add NOT NULL to the column definition. If a column is to be tested for null, use the IS NULL or IS NOT NULL test as in the orders_ship_date_ck check constraint.

check constraints can combine more than one column into a single check constraint as shown below

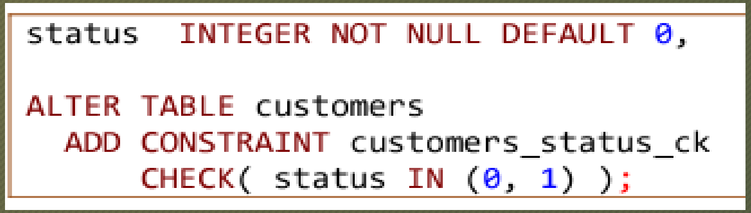


```
ALTER TABLE orders
  ADD CONSTRAINT orders_status_name_ck
      CHECK ( ( status = 'C' OR status = 'I' )
        AND   ( name  <> ' ' ) );
```

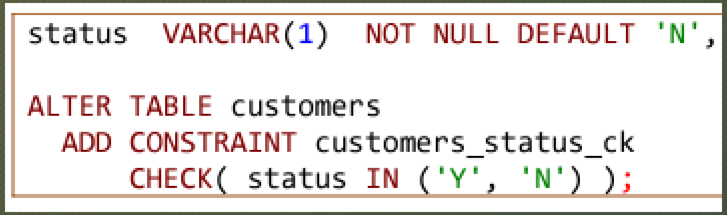
Database Constraints

Representing Boolean Data Types

The Boolean data type is a data type that has one of two possible values, usually denoted as true or false. Not all database management system supports Boolean data type. However Boolean data type can be represented using a CHECK constraint.



```
status   INTEGER NOT NULL DEFAULT 0,

ALTER TABLE customers
  ADD CONSTRAINT customers_status_ck
      CHECK( status IN (0, 1) );
```



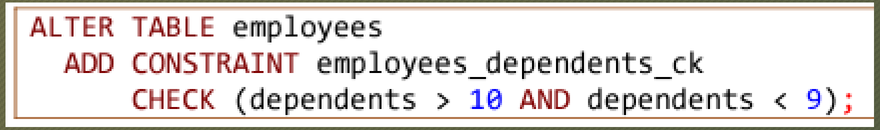
```
status   VARCHAR(1)  NOT NULL DEFAULT 'N',

ALTER TABLE customers
  ADD CONSTRAINT customers_status_ck
      CHECK( status IN ('Y', 'N') );
```

Database Constraints

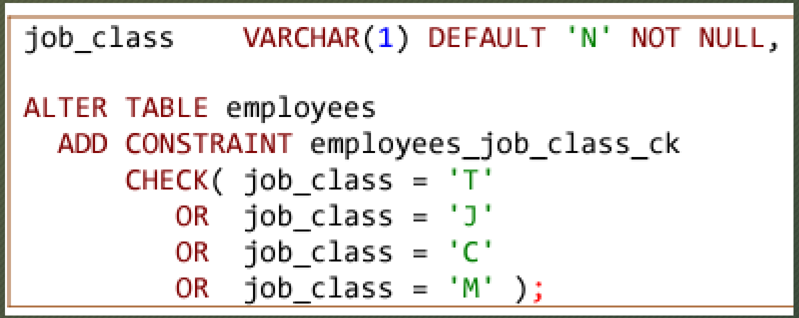
Beware of Semantic with Check Constraint

The DBMS performs no semantic checking on constraints and defaults. It allows the definition of CHECK constraints that contradicts one another. Care must be taken to avoid creating this type of problem

e. g



```
ALTER TABLE employees
  ADD CONSTRAINT employees_dependents_ck
      CHECK (dependents > 10 AND dependents < 9);
```



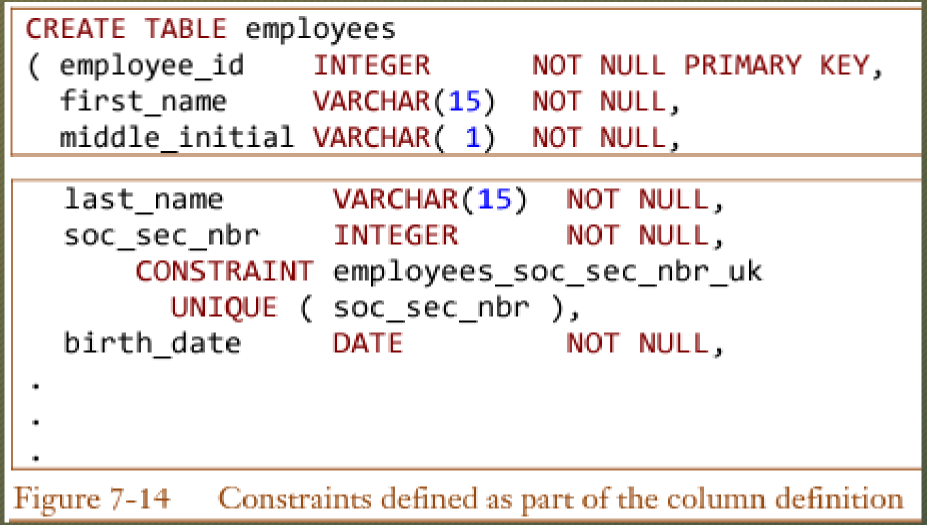
```
job_class    VARCHAR(1) DEFAULT 'N' NOT NULL,

ALTER TABLE employees
  ADD CONSTRAINT employees_job_class_ck
      CHECK( job_class = 'T'
         OR  job_class = 'J'
         OR  job_class = 'C'
         OR  job_class = 'M' );
```

What is the contradiction here ?

Database Constraints

Defining Check Constraints At the Table Level :

Most constraints can be specified as part of a column definition or as a separate clause as shown below.



```
CREATE TABLE employees
( employee_id    INTEGER      NOT NULL PRIMARY KEY,
  first_name     VARCHAR(15)  NOT NULL,
  middle_initial VARCHAR( 1)  NOT NULL,
```

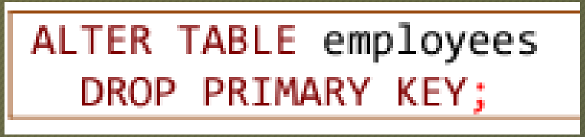
```
  last_name      VARCHAR(15)  NOT NULL,
  soc_sec_nbr    INTEGER      NOT NULL,
      CONSTRAINT employees_soc_sec_nbr_uk
        UNIQUE ( soc_sec_nbr ),
  birth_date     DATE         NOT NULL,
.
.
.
```

Figure 7-14 Constraints defined as part of the column definition

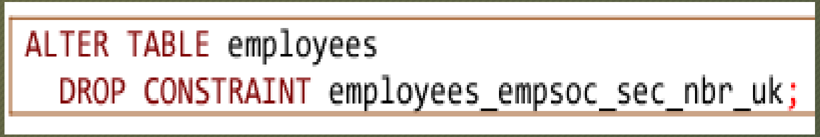
Database Constraints

Adding and Removing Check constraints From Existing Table :

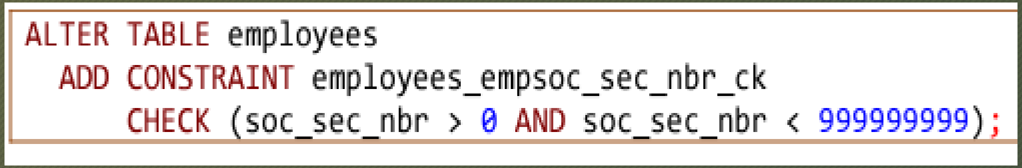
After a table is created , the ALTER TABLE command can be used to add or remove a primary key, unique, foreign key, or check constraint , as the following example illustrate. (To add a constraint ADD CONSTRINT command can be used)



```
ALTER TABLE employees
  DROP PRIMARY KEY;
```



```
ALTER TABLE employees
  DROP CONSTRAINT employees_empsoc_sec_nbr_uk;
```



```
ALTER TABLE employees
  ADD CONSTRAINT employees_empsoc_sec_nbr_ck
      CHECK (soc_sec_nbr > 0 AND soc_sec_nbr < 999999999);
```

Database Constraints

Constraint Unit Testing

Before database is finalized and put into production a strategy must be implemented for testing the database to verify that all constraints meets the business requirements. To accomplish this , it is important to test only one constraint at a time . If two or more constraints are tested at the same time and the test fails , it is difficult sometimes to determine which constraint caused the test to fail

Unit Testing is the process in which the smallest part of an application, one constraint in this case, is independently verified to ensure it performs according to expectations

During the constraint testing

- Column should be tested to ensure they are defined with correct datatype

-Column should be tested to ensure they are accommodate the largest value that might be entered

-Column should be tested to ensure valid data is entered to the database

-Column should be tested to ensure invalid data is rejected

Conclusion