# **Data Manipulation using DML**

## **Objectives**

**4.1 Construct and analyze INSERT statements**

INSERT INTO SELECT, INSERT INTO VALUES

**4.2 Construct and analyze UPDATE statements**

Update data in a single table

**4.3 Construct and analyze DELETE statements**

Delete data in a single table

## **Reading Materials**

SQL Primer - An Accelerated Introduction to SQL Basics: Chapter 6

### Chapter 6: Manipulating Data

#### Overview

In this chapter we study the *Data Manipulation Language (DML)* part of SQL that is used to make changes to the data inside a relational database. The three basic commands of DML are as follows.

* **INSERT -** Populates tables with new data
* **UPDATE -** Updates existing data
* **DELETE -** Deletes data from tables

We have already seen a few examples on the INSERT statement including simple inserts, selective field insertions, and null value inserts. Thus we will concentrate on other ways to use this statement.

#### Inserting Data into a Table from Another Table

You can insert new records into a table from another one by using a combination of INSERT and SELECT. This is pretty close to the way we combined CREATE TABLE and SELECT to create a new table with rows from another table.

Since a query would return you some records, combining it with an insertion command would enter these records into the new table. You can even use a WHERE conditional to limit or filter the records you wish to enter into the new table. We will now create a new table called stdlang\_tbl, which will have only two fields – language and standard. In this we would insert rows from the proglang\_tbl table that have a non-null value in the standard field (Listing 6-1). This will also demonstrate our first use of a boolean operator – NOT.

CREATE TABLE stdlang\_tbl (language varchar(20), standard varchar(10))

INSERT INTO stdlang\_tbl

SELECT language, standard

FROM proglang\_tbl WHERE standard IS NOT NULL

Note that we had to create the table separately in this case and then insert data into it using INSERT and SELECT. The NOT inverts the IS NULL test, that is, if something is a null value instead of normally returning a true, the NOT makes the conditional return a false.

NOT, NULL, True and False

If we consider boolean logic principles, the closest analogue to NULL is the value *false*. However NOT inverts a boolean value: *true* becomes *false*, and *false* becomes *true*.

Testing a value with IS NULL is an SQL comparison, returning a *true* for every null value. This is what we have inverted in the example above using NOT. Don't confuse the meaning of this comparison with what boolean value a NULL closely represents.

When you view the contents of this table, you will notice that it has picked up the languages that actually had a *standard* column value ([Table 6-1](https://viewer.books24x7.com/assetviewer.aspx?bookid=142634&chunkid=694994341&resumebookmarkid=a025dd84-1f41-ee11-aa72-005056b54d63#)).

| language | standard |
| --- | --- |
| Prolog | ISO |
| APL | ANSI |
| JOVIAL | US-DOD |
| APT | ISO |

The data being populated by INSERT INTO and SELECT must adhere to the constraints defined during table creation. If our stdlang\_tbl had defined language as its primary key, our insert statements would run fine unless we encountered a duplicate language value (Listing 6-2). This is not a problem in our case currently since the languages are themselves unique.

DROP TABLE stdlang\_tbl

CREATE TABLE stdlang\_tbl (language varchar(20) PRIMARY KEY,

standard varchar(10))

INSERT INTO stdlang\_tbl

SELECT language, standard

FROM proglang\_tbl WHERE standard IS NOT NULL

What would happen if we somehow violated the constraints? For example, let us go about creating a new table standardizing\_bodies that contains only one field – name. The only constraint on this is UNIQUE. We already know that both Prolog and APT from our proglang\_tbl were standardized by ISO. Let's try to simulate this using code (Listing 6-3).

CREATE TABLE standardizing\_bodies (name varchar(10) UNIQUE)

INSERT INTO standardizing\_bodies

SELECT standard FROM proglang\_tbl WHERE standard IS NOT NULL;

ERROR: duplicate key value violates unique constraint

"standardizing\_bodies\_name\_key"

DETAIL: Key (name)=(ISO) already exists.

Note that the contents of this new table standardizing\_bodies will be empty. Our INSERT operation was a single statement, not a collection of unique inserts. Thus when the constraint was violated, no data was inserted.

#### Updating Existing Data

To modify some data in a record, we use the UPDATE command. While it cannot add or delete records (those responsibilities are delegated to other commands), if a record exists it can modify its data even affecting multiple fields in one go and applying conditions. The general syntax of an UPDATE statement is given below (Listing 6-4).

UPDATE <table\_name> SET

<column1> = <value>,

<column2> = <value>,

<column3> = <value>

...

WHERE <condition>

Let us now return to our *proglang\_tbl* table and add a new row about the *Forth* and *Tcl* programming languages ([Listing 6-5](https://viewer.books24x7.com/assetviewer.aspx?bookid=142634&chunkid=251406291&resumebookmarkid=a025dd84-1f41-ee11-aa72-005056b54d63#)).

INSERT INTO proglang\_tbl (id, language, author, year, standard) VALUES

(6, 'Forth', 'Moore', 1973, NULL);

INSERT INTO proglang\_tbl (id, language, author, year, standard) VALUES

(7, 'Tcl', 'Ousterhout', 1988, NULL);

What if we suddenly wanted to add 10 years to each language's creation year? Since we want to apply the UPDATE logic to every row, we can forego the search conditions (Listing 6-6).

UPDATE proglang\_tbl SET year = year + 10

This query would increase all language creation years by 10. There is no ambiguity here, since the right-hand side year + 10 is calculated first and then assigned to the year field. This happens for all rows. To get back to our original dates, simply run the same query with the SET column as year = year – 10.

We later realize that the Forth language was created near 1972 (instead of 1973), and it actually has been standardized in 1994 by the ANSI. Thus we now go about correcting our mistakes by writing our update queries to reflect this data ([Listing 6-7](https://viewer.books24x7.com/assetviewer.aspx?bookid=142634&chunkid=251406291&resumebookmarkid=a025dd84-1f41-ee11-aa72-005056b54d63#)). We should note that we must include a search condition for the Forth language only.

UPDATE proglang\_tbl SET year = 1972 WHERE language = 'Forth'

UPDATE proglang\_tbl SET standard = 'ANSI' WHERE language = 'Forth'

We could have easily combined updating the multiple fields in a single statement, thus saving the DBMS engine the trouble to find the row again (Listing 6-8).

UPDATE proglang\_tbl SET year = 1972, standard = 'ANSI' WHERE language = 'Forth'

If you've typed the statement correctly and no errors are thrown back, the contents of the record in question would have been modified as intended. Verifying the result of the same involves a simple query the likes of which we have seen in previous examples.

#### Deleting Data from Tables

You can use the DELETE command to delete records from a table. This means that you can choose which records you want to delete based on a condition or delete all records, but you cannot delete certain fields of a record using this statement. The general syntax of the DELETE statement is given below (Listing 6-9).

DELETE FROM <table\_name> WHERE <condition>

While putting a conditional clause in the DELETE is optional, it is almost always used – simply because not using it would cause all the records to be deleted from a table, which is a rarely valid need. Luckily, we have a spare table stdlang\_tbl that is not needed anymore, so let's try deleting all rows from it (Listing 6-10).

DELETE FROM stdlang\_tbl;

If we try to verify contents of this table, we'd get no data rows back. Only the column headers would be visible.

| language | standard |
| --- | --- |

We now write the full statement to delete the record corresponding to Forth from the table. Again, we will have to include the search condition in the WHERE clause ([Listing 6-11](https://viewer.books24x7.com/assetviewer.aspx?bookid=142634&chunkid=416452970&resumebookmarkid=a025dd84-1f41-ee11-aa72-005056b54d63#), [Table 6-2](https://viewer.books24x7.com/assetviewer.aspx?bookid=142634&chunkid=416452970&resumebookmarkid=a025dd84-1f41-ee11-aa72-005056b54d63#)).

DELETE FROM proglang\_tbl WHERE language = 'Forth';

| id | language | author | year | standard |
| --- | --- | --- | --- | --- |
| 1 | Prolog | Colmerauer | 1972 | ISO |
| 2 | Perl | Wall | 1987 |  |
| 3 | APL | Iverson | 1964 | ANSI |
| 4 | JOVIAL | Schwartz | 1959 | US-DOD |
| 5 | APT | Ross | 1959 | ISO |
| 7 | Tcl | Ousterhout | 1988 |  |

You should always be careful about the WHERE clauses you put on a DELETE statement. They should never be too broad, lest you end up deleting more data than you intended.

-----------------------------------------------------------------------------------------

## Domain

### Pre-Assessment

Q1: Copying Records to a New Table (ID 27896) = V

What statement is used to copy records from an existing table to a new table?

SELECT

SELECT INTO

SELECT WITHIN

INSERT INTO SELECT

**SELECT INTO**

Q2: INSERT INTO…VALUES Statements (ID 27897) = V

What must be included in parentheses in an INSERT INTO…VALUES statement?

Required fields

Column types

Number of columns

Null values

**Required fields**

Q3: UPDATE Statements (ID 27898) = V

What clause should always be included in an UPDATE statement?

GROUP BY

FROM

WHERE

HAVING

**WHERE**

Q4: DELETE Statements (ID 27899) = V

Before running a DELETE statement, it is always a good idea to run a(n) \_\_\_ statement on the table that is about to be changed.

DROP

SELECT

INSERT

UPDATE

**SELECT**

### Post-Assessment

Q1: Copying Records to an Existing Table (ID 27900) = V

What statement is used to copy records from one table into an existing table?

SELECT

SELECT INTO

SELECT WITHIN

INSERT INTO SELECT

**INSERT INTO SELECT**

Q2: Inserting Values Into a Table (ID 27901) = V

What statement is used to insert values into a table?

ADD VALUES

INSERT INTO…VALUES

INSERT VALUES…INTO

ADD INTO…VALUES

**INSERT INTO…VALUES**

Q3: Changing Data (ID 27902) = V

What statement is used to change existing data?

NEW

CHANGE

MODIFY

UPDATE

**UPDATE**

Q4: Removing Records (ID 27903) = V

What keyword removes records from a table?

DELETE

DROP

REMOVE

CANCEL

**DELETE**

Q5: Deleting Data (ID 27904) = V

What is the difference between a DELETE statement and a TRUNCATE TABLE statement?

One can identify specific records that are being deleted with a DELETE statement, while a TRUNCATE TABLE statement deletes all data in a table

One can identify specific records that are being deleted with a TRUNCATE TABLE statement, while a DELETE statement deletes all data in a table

One can identify specific records that are being deleted with a DELETE statement, while a TRUNCATE TABLE statement allows one to delete specific characters in a record

One can identify specific records that are being deleted with a TRUNCATE TABLE statement, while a DELETE statement allows one to delete specific characters in a record

**One can identify specific records that are being deleted with a TRUNCATE TABLE statement, while a DELETE statement deletes all data in a table**

**One can identify specific records that are being deleted with a DELETE statement, while a TRUNCATE TABLE statement deletes all data in a table**

## Videos

### Using Queries with the UPDATE Statement (6m)

– - Using a subquery to update product prices based on the supplier

UPDATE Products

SET UnitPrice = (UnitPrice \* 1.1)

WHERE ProductID IN

(SELECT ProductID

FROM Products INNER JOIN Suppliers

ON Products.SupplierID = Suppliers.SupplierID

WHERE Suppliers.CompanyName = ‘Exotic Liquids’)

– - Verify records

SELECT \* FROM Suppliers

- - Select all products ordered by SupplierID

SELECT \* FROM Products ORDER BY SupplierID

### Using Queries with INSERT Statement (5m)

- - IDENTITY (1,1) is similar to AUTO INCREMENT (start at 1 and increase by 1 for each record in a table)

CREATE TABLE Customers2

(

CustomerID int IDENTITY (1,1) NOT NULL,

Name varchar(25) NOT NULL

)

CREATE TABLE Orders2

(

OrderID int IDENTITY (1,1) NOT NULL,

CustomerID int NOT NULL,

OrderDate datetime NOT NULL

)

INSERT INTO Customers2 VALUES (‘John Smith’)

SELECT \* FROM Customers2

INSERT INTO Orders2

SELECT CustomerID, GETDATE()

FROM Customers2 WHERE Name = ‘John Smith’

SELECT \* FROM Orders2

### Using Queries with DELETE FROM Statement (4m)

DELETE FROM Products

WHERE ProductID IN

(SELECT ProductID

FROM Products INNER JOIN Suppliers

ON Products.SupplierID = Suppliers.SupplierID

WHERE Suppliers.CompanyName = ‘Exotic Liquids’)

## Assignment

Please upload screenshots of your solutions for the Exercise

## Quiz

Q1: Given the Books table shown below:



You wish to copy the records from the Books table to the Audio\_Books table.

Assume that the Audio\_Books table already exists and that the data types in the source and target tables match.

Using the dropdown menus, fill in the blanks to complete the SQL statement shown below.

\_\_\_ \_\_\_ (ISBN, title, author, price, year\_of\_release)

\_\_\_ ISBN, title, author, price, year\_of\_release FROM \_\_\_

**INSERT INTO Audiobooks (ISBN, title, author, price, year\_of\_release)**

**SELECT ISBN, title, author, price, year\_of\_release FROM Books**

Q2: What statement is used to copy records from one table into an existing table?

INSERT INTO SELECT

SELECT INTO

SELECT

SELECT WITHIN

**INSERT INTO SELECT**

Q3: Given the Audio\_Books table shown below:



Which of the following statements can be used to remove all records from the table, but not the table itself?

Choose all that apply.

DELETE FROM Audio\_Books;

DROP TABLE Audio\_Books;

DELETE \* FROM Audio\_Books;

TRUNCATE TABLE Audio\_Books;

**DELETE FROM Audio\_Books;**

**TRUNCATE TABLE Audio\_Books;**

Q4: You have been asked to write a database query that copies records from one table to another table. Specifically, you need to copy records from the Employee table to the Retired\_Employees table.

You create the Retired\_Employees table and specify the same number of columns, data types and primary key as that of the Employee table. The primary key in the table is empID.

There are two records in the Employee table as shown below.



You execute the following INSERT INTO SELECT statement:

INSERT INTO Retired\_Employees (empID, fName, lName, age, empAddress, empEmail, empPhoneNo, salary)

SELECT empID, fName, lName, age, empAddress, empEmail, empPhoneNo, salary FROM Employee;

By accident, you run the INSERT INTO SELECT statement separately again.

How many records will there be in the Retired\_Employees table?

6

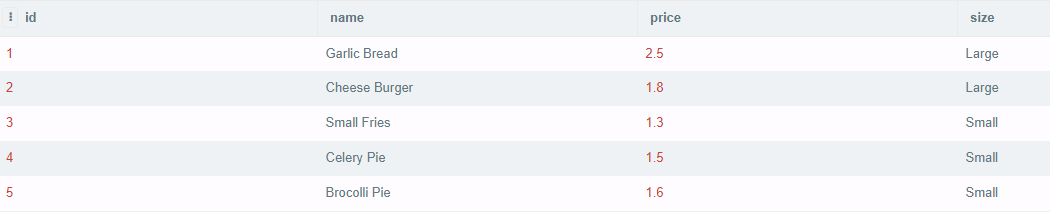
2

4

0

**2**

Q5: Given the following table called Meals, in which the primary key is the id field.



Given another table called Low\_Price\_Meals, which has the same structure as the Meals table (including the same primary key field).

What will be the result of running the following query?

INSERT INTO Low\_Price\_Meals (id, name, price)

SELECT id, name, price from Meals WHERE price <= 1.50

1 record will be copied to the Low\_Price\_Meals table.

Due to a syntax error, the INSERT INTO SELECT statement will fail to execute.

2 records will be copied to the Low\_Price\_Meals table.

None of the other listed answers are correct.

5 records will be copied to the Low\_Price\_Meals table.

**2 records will be copied to the Low\_Price\_Meals table.**

Q6: What keyword removes records from a table?

REMOVE

DROP

CANCEL

DELETE

**DELETE**

Q7: What is the difference between a DELETE statement and a TRUNCATE TABLE statement?

One can identify specific records that are being deleted with a DELETE statement, while a TRUNCATE TABLE statement deletes all data in a table

One can identify specific records that are being deleted with a TRUNCATE TABLE statement, while a DELETE statement deletes all data in a table

One can identify specific records that are being deleted with a DELETE statement, while a TRUNCATE TABLE statement allows one to delete specific characters in a record

One can identify specific records that are being deleted with a TRUNCATE TABLE statement, while a DELETE statement allows one to delete specific characters in a record

**One can identify specific records that are being deleted with a DELETE statement, while a TRUNCATE TABLE statement deletes all data in a table**

Q8: What statement is used to insert values into a table?

INSERT INTO

ADD INTO…VALUES

INSERT VALUES…INTO

ADD VALUES

**INSERT INTO**

Q9: What statement is used to change existing data?

NEW

CHANGE

UPDATE

MODIFY

**UPDATE**

Q10: Given the following table called Audio\_Books:



Which of the following statements can be used to change the value of the field, 'year\_of\_release' for the audio book with the title of 'The Lincoln Highway' to 2021-05-30?

UPDATE Audio\_Books set year\_of\_release = 2021-05-30 WHERE title = The Lincoln Highway;

MODIFY Audio\_Books set year\_of\_release = '2021-05-30' WHERE title = 'The Lincoln Highway';

UPDATE Audio\_Books set year\_of\_release = '2021-05-30' WHERE title = 'The Lincoln Highway';

UPDATE Audio\_Books set 'year\_of\_release' = '2021-05-30' WHERE 'title' = 'The Lincoln Highway';

**UPDATE Audio\_Books set year\_of\_release = '2021-05-30' WHERE title = 'The Lincoln Highway';**

Q11: Given the table, Audio\_Books shown below:



Which of the following statements can be used to remove the record from the table with a title of 'One Step Too Far'?

Choose all that apply.

DELETE FROM Audio\_Books WHERE title = 'One Step Too Far';

REMOVE FROM Audio\_Books WHERE title = 'One Step Too Far';

DELETE WHERE title = 'One Step Too Far' FROM Audio\_Books;

TRUNCATE FROM Audio\_Books WHERE title = 'One Step Too Far';

**DELETE FROM Audio\_Books WHERE title = 'One Step Too Far';**

Q12: The INSERT INTO SELECT statement requires that the data types in source and target tables match. Is this statement true or false?

**True**