Data Manipulation Language (DML) in SQL

Hia Al Saleh

October 7th, 2024

Contents

1 Introduction to DML

Data Manipulation Language (DML) is a subset of SQL commands used to manage data in database tables. The primary DML commands are:

- INSERT - Adds new records to a table.
- UPDATE - Modifies existing records.
- DELETE - Removes records from a table.

DML commands require knowledge of:

- Table column names and data types
- Whether columns are primary keys, unique, or allow NULLs
- Table and column constraints

2 INSERT Command

The INSERT command adds new rows to a table. It can be executed in several ways:

2.1 Basic Syntax

```
INSERT INTO tablename (column1, column2, column3, ...)
VALUES (value1, value2, value3, ...);
```

The order and number of columns must match between the two lists. Each value must have a compatible data type with its respective column.

2.2 Insert by Column Positions

An INSERT without specifying column names relies on column order:

```
INSERT INTO tablename VALUES (value1, value2, value3, ...);
```

This method is generally discouraged, as changes in the table structure may cause it to fail.

2.3 Insert from Another Table

The INSERT INTO ... SELECT syntax adds data from another table:

```
INSERT INTO tablename
SELECT column1, column2
FROM another_table
WHERE condition;
```

2.4 Example

```
CREATE TABLE teams (
    team_id INTEGER NOT NULL PRIMARY KEY,
    team_name VARCHAR(30) NOT NULL,
    city VARCHAR(20) NOT NULL,
    conference VARCHAR(10) NOT NULL
);

INSERT INTO teams (team_id, team_name, city, conference)
VALUES (1, 'Red Wings', 'Detroit', 'Eastern');
```

3 UPDATE Command

The UPDATE command modifies existing records in a table. It can update all rows or a subset of rows based on a condition.

```
UPDATE tablename
SET column1 = value1, column2 = value2
WHERE condition;
```

Using a WHERE clause restricts updates to specific rows.

3.1 Example with WHERE

```
UPDATE superheroes
SET secret_identity = 'Diana Prince'
WHERE hero_id = 3;
```

3.2 Using Comparison Operators

Comparison operators such as =, <> (not equal), <, >, <=, and >= can be used in the WHERE clause to define conditions. Example:

```
UPDATE personnel
SET salary = salary * 1.07
WHERE jobgrade <= 4;
```

3.3 Pattern Matching with LIKE

The LIKE operator uses wildcards for pattern matching:

```
UPDATE superheroes
SET gender = 'female'
WHERE hero_name LIKE '%Woman';
```

Here, % matches any sequence of characters, while _ matches a single character.

4 DELETE Command

The DELETE command removes rows from a table. It does not require column names, as it removes entire rows.

```
DELETE FROM tablename
WHERE condition;
```

A WHERE clause is essential to avoid deleting all rows.

4.1 Example

```
DELETE FROM superheroes
WHERE hero_id > 3;
```

5 Preserving Referential Integrity with Foreign Keys

When inserting, updating, or deleting rows in tables with foreign key relationships, referential integrity must be maintained:

- **Inserting a row** in a foreign-key table requires the foreign key value to match a primary key in the parent table.
- **Updating a row** in the foreign-key table must ensure the updated foreign key matches an existing primary key in the parent table.
- **Deleting a row** in the parent table may not be allowed if foreign keys in child tables reference that row.

6 Auto-Incrementing IDs

Primary keys can be set to auto-increment, automatically generating unique values:

```
CREATE TABLE counting (
    id INTEGER AUTO_INCREMENT NOT NULL PRIMARY KEY,
    name VARCHAR(10)
);
INSERT INTO counting (name) VALUES ('first');
```

Auto-increment values continue to increase even if rows are deleted.

7 Examples of DML Commands with Constraints

- **Creating Tables with Foreign Keys**

```
CREATE TABLE authors (
    id INTEGER AUTO_INCREMENT NOT NULL PRIMARY KEY,
    fname VARCHAR(20) NOT NULL,
    lname VARCHAR(25) NOT NULL
);

CREATE TABLE books (
    id INTEGER AUTO_INCREMENT NOT NULL PRIMARY KEY,
    title VARCHAR(25),
    author_id INTEGER NOT NULL,
    CONSTRAINT author_fk FOREIGN KEY (author_id)
    REFERENCES authors(id)
);
```

- **Inserting Data with Foreign Key Constraint**

```
INSERT INTO books (title, author_id) VALUES ('Watchers', 1);
```