Chapter 27: MERGE

MERGE (often also called UPSERT for "update or insert") allows to insert new rows or, if a row already exists, to update the existing row. The point is to perform the whole set of operations atomically (to guarantee that the data remain consistent), and to prevent communication overhead for multiple SQL statements in a client/server system.

Section 27.1: MERGE to make Target match Source

```
MERGE INTO targetTable t
    USING sourceTable s
        ON t.PKID = s.PKID
    WHEN MATCHED AND NOT EXISTS (
            SELECT s.ColumnA, s.ColumnB, s.ColumnC
            INTERSECT
            SELECT t.ColumnA, t.ColumnB, s.ColumnC
            )
        THEN UPDATE SET
            t.ColumnA = s.ColumnA
            ,t.ColumnB = s.ColumnB
            ,t.ColumnC = s.ColumnC
    WHEN NOT MATCHED BY TARGET
        THEN INSERT (PKID, ColumnA, ColumnB, ColumnC)
        VALUES (s.PKID, s.ColumnA, s.ColumnB, s.ColumnC)
    WHEN NOT MATCHED BY SOURCE
        THEN DELETE
    ;
```

Note: The `AND NOT EXISTS` portion prevents updating records that haven't changed. Using the `INTERSECT` construct allows nullable columns to be compared without special handling.

Section 27.2: MySQL: counting users by name

Suppose we want to know how many users have the same name. Let us create table `users` as follows:

```
create table users(
    id int primary key auto_increment,
    name varchar(8),
    count int,
    unique key name(name)
);
```

Now, we just discovered a new user named Joe and would like to take him into account. To achieve that, we need to determine whether there is an existing row with his name, and if so, update it to increment count; on the other hand, if there is no existing row, we should create it.

MySQL uses the following syntax : insert … on duplicate key update …. In this case:

```
insert into users(name, count)
       values ('Joe', 1)
       on duplicate key update count=count+1;
```

Section 27.3: PostgreSQL: counting users by name

Suppose we want to know how many users have the same name. Let us create table `users` as follows:

```
create table users(
    id serial,
    name varchar(8) unique,
    count int
);
```

Now, we just discovered a new user named Joe and would like to take him into account. To achieve that, we need to determine whether there is an existing row with his name, and if so, update it to increment count; on the other hand, if there is no existing row, we should create it.

PostgreSQL uses the following syntax : insert … on conflict … do update …. In this case:

```
insert into users(name, count)
    values('Joe', 1)
    on conflict (name) do update set count = users.count + 1;
```