REVERSE ENGINEERING

H2 DATABASE
RUNTIME TERROR

Pull Request

We have submitted a pull request for the issue:
https://github.com/h2database/h2database/issues/2407 and the pull request is

https://github.com/h2database/h2database/pull/2504

Issue

When the user ran the following sequence of statements.

```
CREATE TABLE TEST(A CHAR(2), B CHAR(2));
INSERT INTO TEST VALUES ('a ', 'b ');
SELECT A || B FROM TEST;
```

It gave the output as “ab” with the right padding removed.

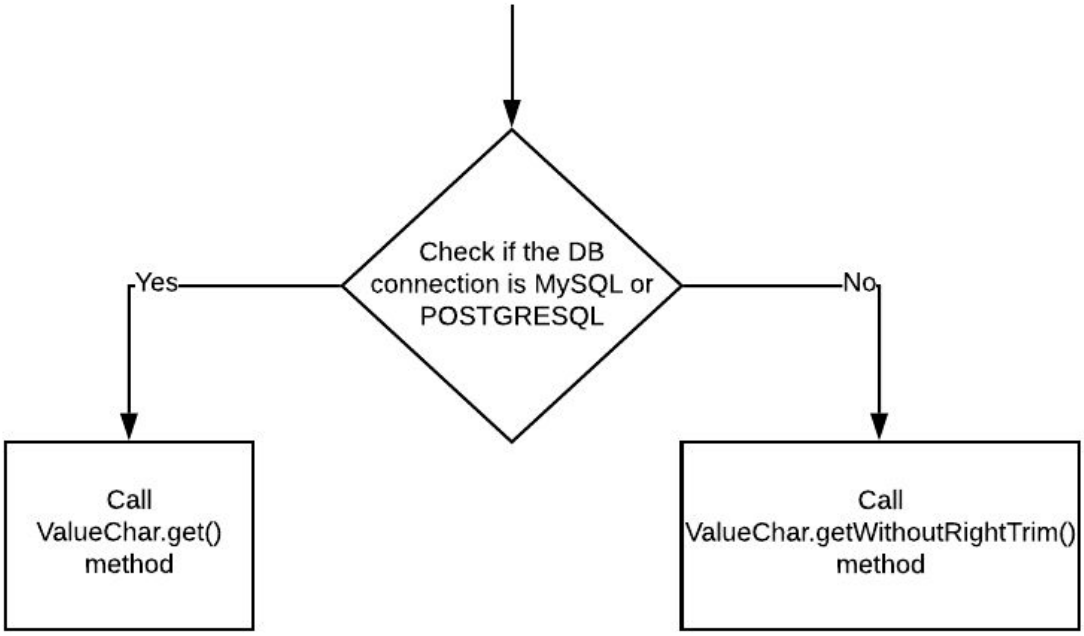
As pointed out by the issue reporter, right padding must be removed only in MySQL, PostgreSQL and for DB2, Firebird, HSQLDB, Oracle, SQLite, SQL Server, and H2 modes it should not be removed. Currently, this is not working in the H2 mode.

After close observation and debugging we realized the problem lied while inserting the values in the table when the values consisted of right whitespaces.

Approach

We first tested the query on the H2 console and observed that the issue is not with the String concatenation, but with the actual insert itself. For example, strings like “a “are getting stored in the database as “a” by removing the trailing whitespace. As we were

already familiar with the code base, we knew we should start looking from the Insert.java class which handles the INSERT statement. On debugging the issue further we found that the get() method in the ValueChar class (implementation of CHAR data type) has a call to trimRight() method in the same class that basically trims the whitespaces from the end of the string. So we created another function called getWithoutRightTrim() in the ValueChar class that doesn't have a call to the trimRight() method. In the method that calls the ValueChar.get() method, we first included the code to check the existing database connection from the database URL and if it is MySQL or POSTGRESQL, it will call the ValueChar.get() method, else for all the other connections we will call the ValueChar.getWithoutRightTrim() function.



Experience

We understood better how the insert query works internally ie how the values are actually getting added into the respective columns. We also got to know that before adding the values into the columns, several checks/preprocessing steps are done like getting the count

of trailing/leading whitespaces, trimming the whitespaces, checking the length of the provided values with the length of the target types (columns), etc.

After we submitted the pull request, the key developer gave some suggestions like avoid using the Session object in the Value class like we did to check the database connection URL as it violates the architecture, and need to make some more adjustments for PostgreSQL while returning the result. We are taking her suggestions into consideration and are making changes as she suggested.