Full Stack
Web Programming

SQL Commands: Part 2

Lesson 8

Lesson outline

- Introduction
- DQL – Data Query Language
- DML – Data Manipulation Language

Data Query Language (DQL)

It uses only one command:

SELECT - is used to retrieve data from the a database.

Data Query Language (DQL)

DQL statements are used for performing queries on the data within schema objects. The purpose of DQL Command is to get some schema relation based on the query passed to it.

DQL - SQL: SELECT

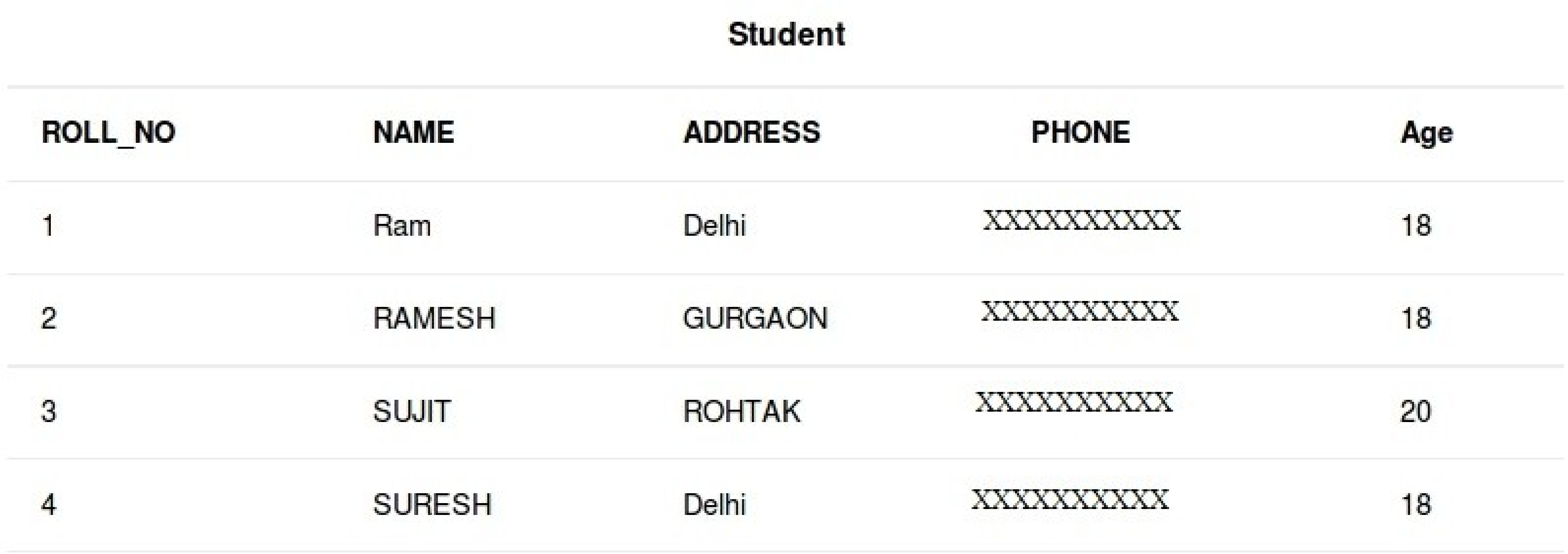
Select is the most commonly used statement in SQL. The SELECT Statement in SQL is used to retrieve or fetch data from a database. We can fetch either the entire table or according to some specified rules. The data returned is stored in a result table. This result table is also called result-set.

With the SELECT clause of a SELECT command statement, we specify the columns that we want to be displayed in the query result and, optionally, which column headings we prefer to see above the result table.

The select clause is the first clause and is one of the last clauses of the select statement that the database server evaluates. The reason for this is that before we can determine what to include in the final result set, we need to know all of the possible columns that could be included in the final result set.

DQL - SQL: SELECT

Sample Table:

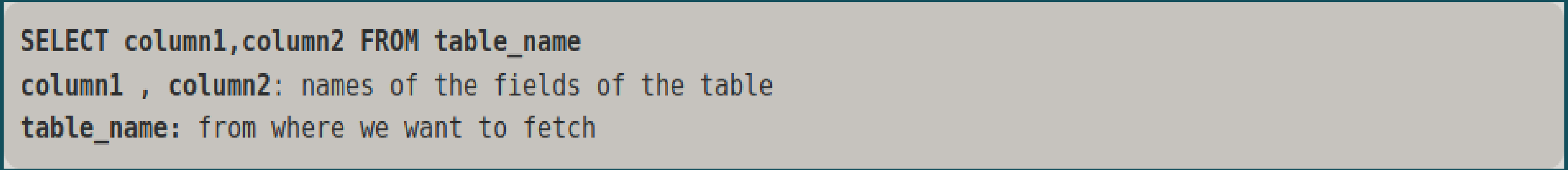


Student

ROLL_NO	NAME	ADDRESS	PHONE	Age
1	Ram	Delhi	XXXXXXXXXX	18
2	RAMESH	GURGAON	XXXXXXXXXX	18
3	SUJIT	ROHTAK	XXXXXXXXXX	20
4	SURESH	Delhi	XXXXXXXXXX	18

DQL - SQL: SELECT

Basic Syntax:

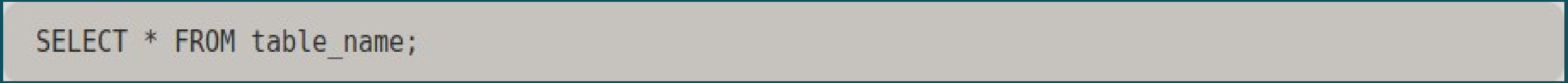


```
SELECT column1,column2 FROM table_name
column1 , column2: names of the fields of the table
table_name: from where we want to fetch
```

This query will return all the rows in the table with fields column1 , column2.

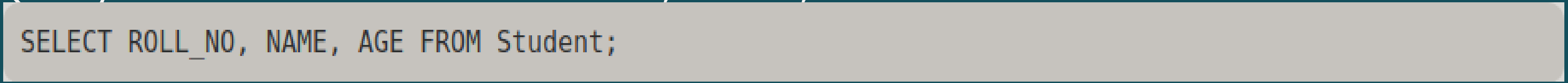
DQL - SQL: SELECT

To fetch the entire table or all the fields in the table:



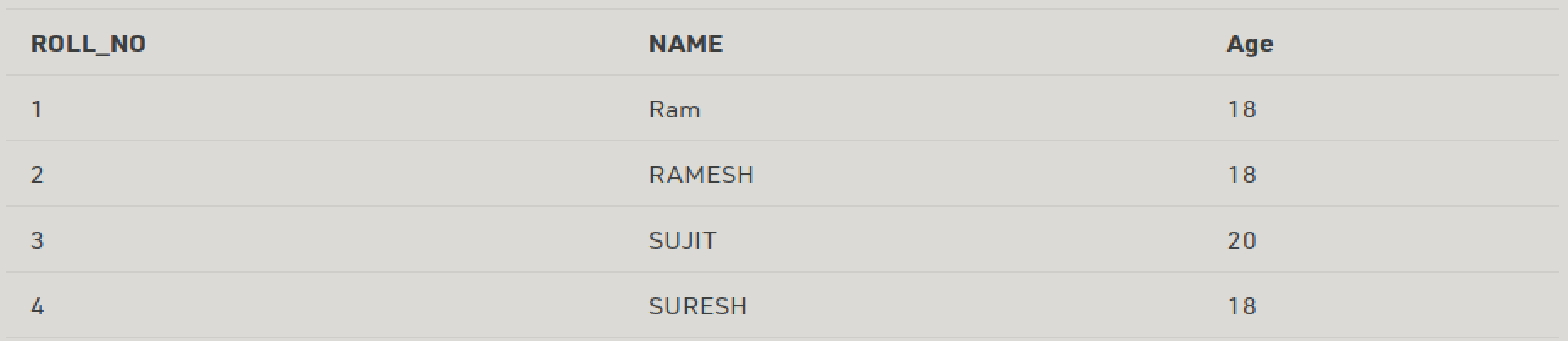
```
SELECT * FROM table_name;
```

Query to fetch the fields ROLL_NO, NAME, AGE from the table Student:



```
SELECT ROLL_NO, NAME, AGE FROM Student;
```

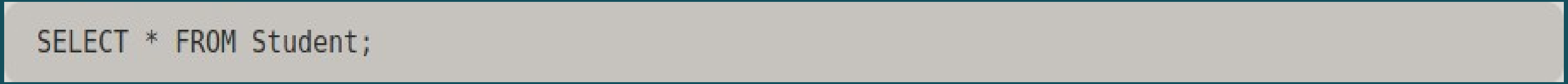
Output:



ROLL_NO	NAME	Age
1	Ram	18
2	RAMESH	18
3	SUJIT	20
4	SURESH	18

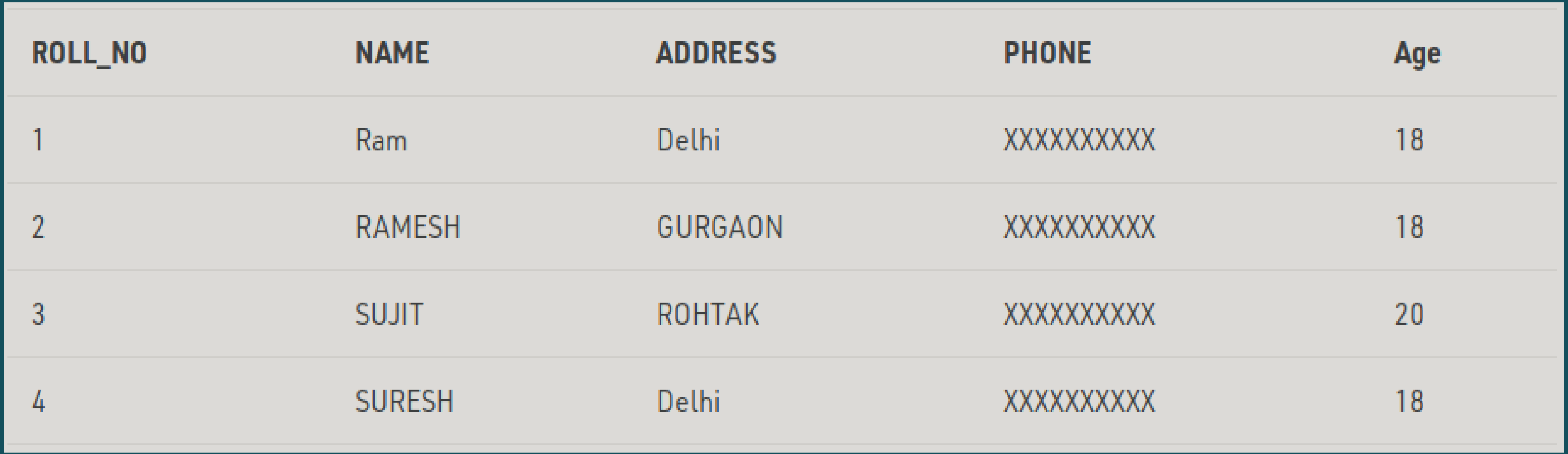
DQL - SQL: SELECT

To fetch all the fields from the table Student:



```
SELECT * FROM Student;
```

Output:



ROLL_NO	NAME	ADDRESS	PHONE	Age
1	Ram	Delhi	XXXXXXXXXX	18
2	RAMESH	GURGAON	XXXXXXXXXX	18
3	SUJIT	ROHTAK	XXXXXXXXXX	20
4	SURESH	Delhi	XXXXXXXXXX	18

Data Manipulation Language (DML)

The SQL commands that deals with the manipulation of data present in the database belong to DML or Data Manipulation Language and this includes most of the SQL statements.

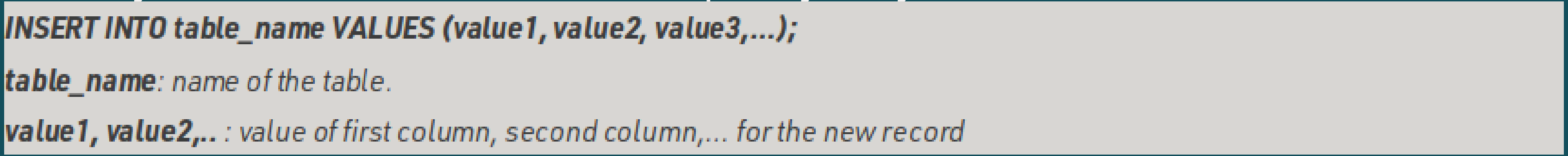
DML commands are used to modify the database. It is responsible for all form of changes in the database.
The command of DML is not auto-committed that means it can't permanently save all the changes in the database. They can be rollback.

Examples of DML:
INSERT – is used to insert data into a table.
UPDATE – is used to update existing data within a table.
DELETE – is used to delete records from a database table.

DML - SQL: INSERT INTO Statement

The INSERT INTO statement of SQL is used to insert a new row in a table. There are two ways of using INSERT INTO statement for inserting rows:

1. Only values: First method is to specify only the value of data to be inserted

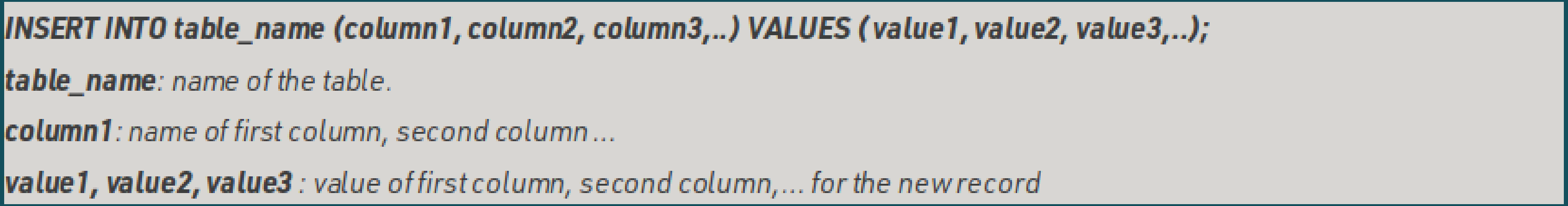


```
INSERT INTO table_name VALUES (value1, value2, value3,...);
```

table_name: *name of the table.*

value1, value2,.. : *value of first column, second column,... for the new record*

2. Column names and values both: In the second method we will specify both the columns which we want to fill and their corresponding values as shown below:



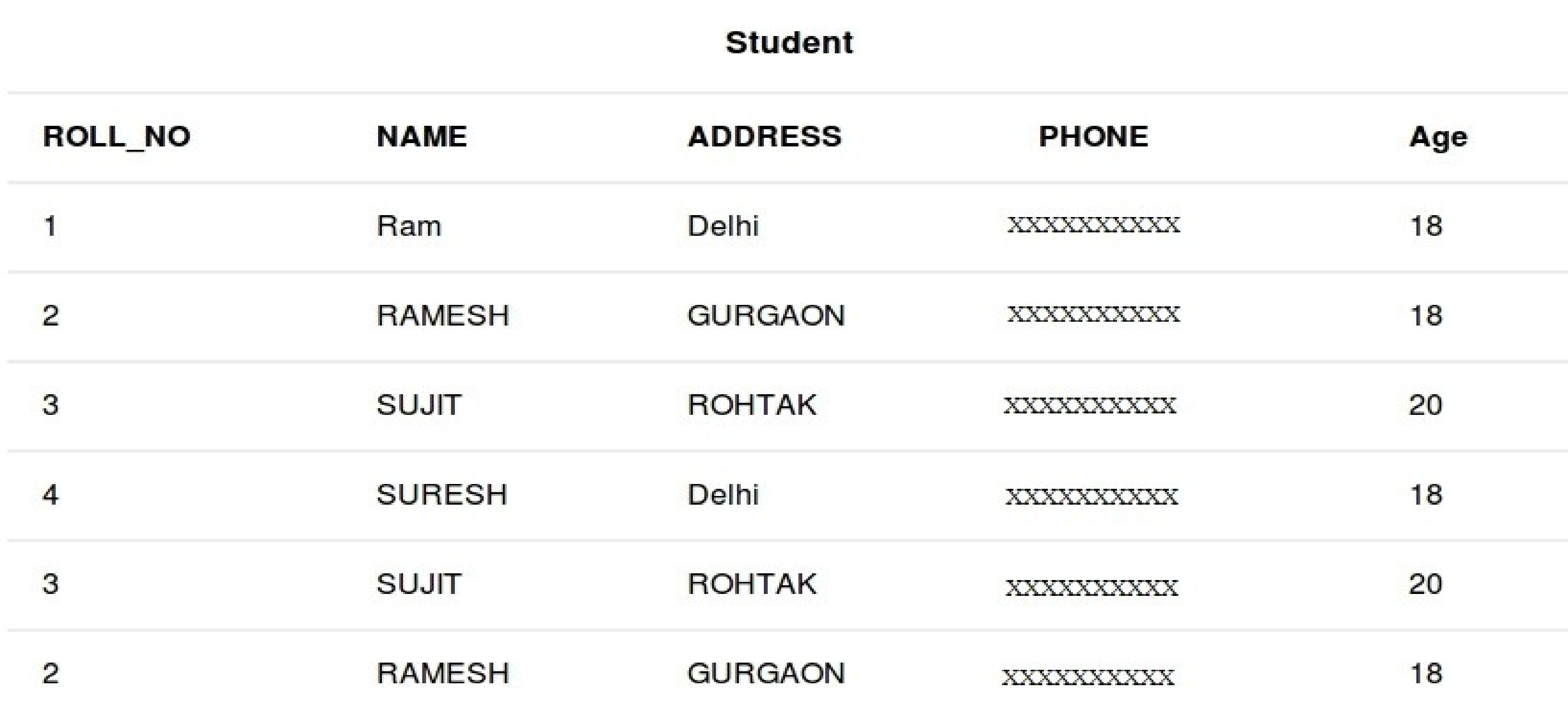
```
INSERT INTO table_name (column1, column2, column3,..) VALUES ( value1, value2, value3,..);
```

table_name: *name of the table.*

column1: *name of first column, second column ...*

value1, value2, value3 : *value of first column, second column,... for the new record*

DML - SQL: INSERT INTO Statement



Student

ROLL_NO	NAME	ADDRESS	PHONE	Age
1	Ram	Delhi	XXXXXXXXXX	18
2	RAMESH	GURGAON	XXXXXXXXXX	18
3	SUJIT	ROHTAK	XXXXXXXXXX	20
4	SURESH	Delhi	XXXXXXXXXX	18
3	SUJIT	ROHTAK	XXXXXXXXXX	20
2	RAMESH	GURGAON	XXXXXXXXXX	18

DML - SQL: INSERT INTO Statement

Notice that the columns for which the values are not provided are filled by null. Which is the default values for those columns.

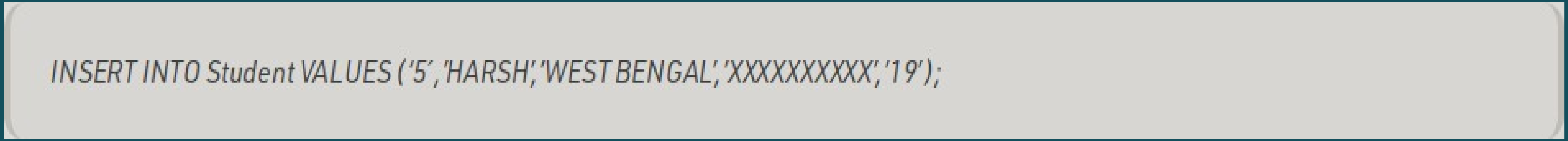
Using SELECT in INSERT INTO Statement

We can use the SELECT statement with INSERT INTO statement to copy rows from one table and insert them into another table.The use of this statement is similar to that of INSERT INTO statement. The difference is that the SELECT statement is used here to select data from a different table. The different ways of using INSERT INTO SELECT statement are shown below:

Inserting all columns of a table: We can copy all the data of a table and insert into in a different table

DML - SQL: INSERT INTO Statement

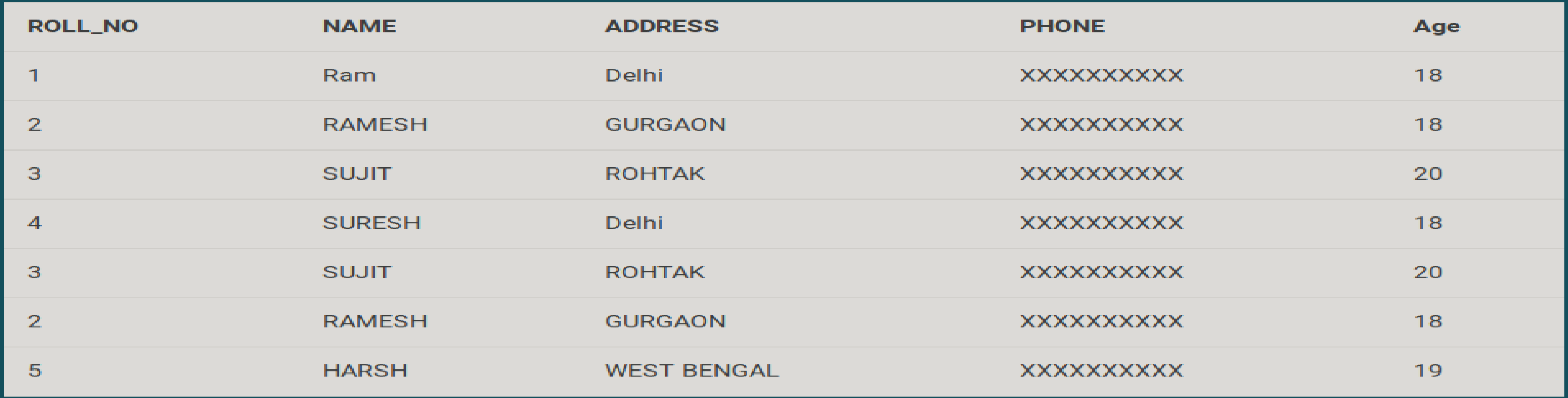
Method 1 (Inserting only values) :



```
INSERT INTO Student VALUES (‘5’,’HARSH’,’WEST BENGAL’,’XXXXXXXXXX’,’19’);
```

Output:

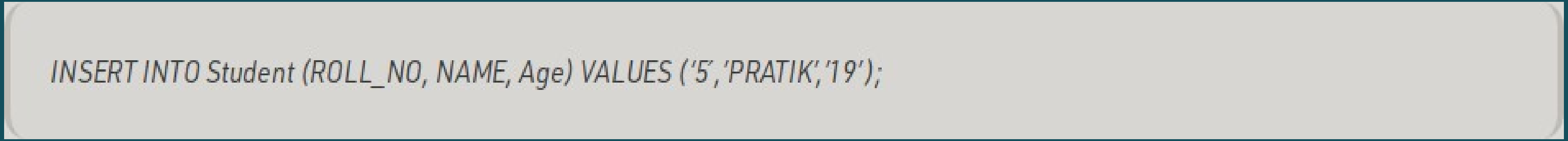
The table Student will now look like:



ROLL_NO	NAME	ADDRESS	PHONE	Age
1	Ram	Delhi	XXXXXXXXXX	18
2	RAMESH	GURGAON	XXXXXXXXXX	18
3	SUJIT	ROHTAK	XXXXXXXXXX	20
4	SURESH	Delhi	XXXXXXXXXX	18
3	SUJIT	ROHTAK	XXXXXXXXXX	20
2	RAMESH	GURGAON	XXXXXXXXXX	18
5	HARSH	WEST BENGAL	XXXXXXXXXX	19

DML - SQL: INSERT INTO Statement

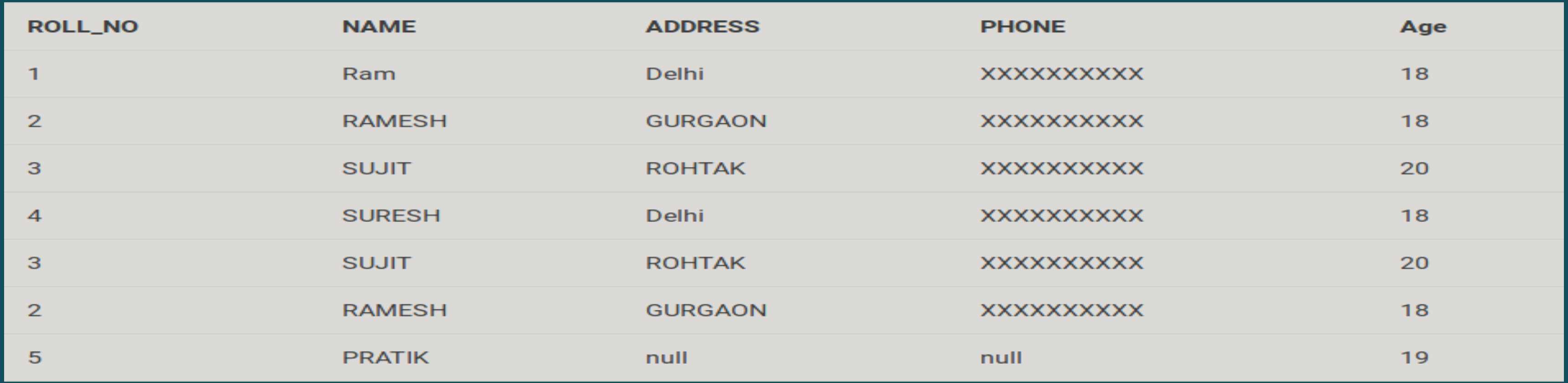
Method 2 (Inserting values in only specified columns):



```
INSERT INTO Student (ROLL_NO, NAME, Age) VALUES (‘5’,‘PRATIK’,‘19’);
```

Output:

The table Student will now look like:

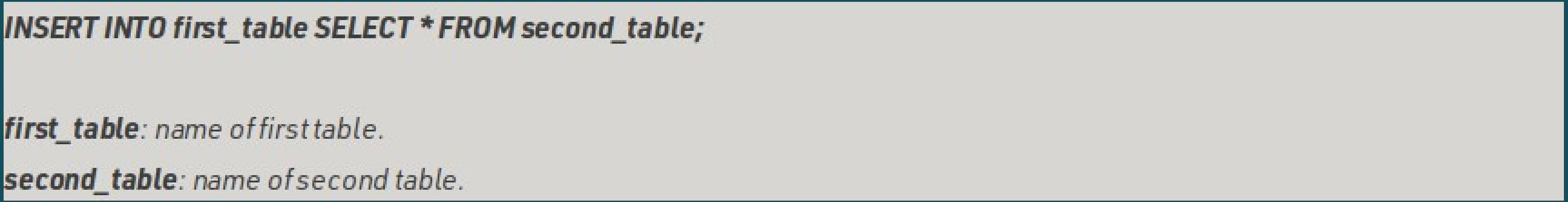


ROLL_NO	NAME	ADDRESS	PHONE	Age
1	Ram	Delhi	XXXXXXXXXX	18
2	RAMESH	GURGAON	XXXXXXXXXX	18
3	SUJIT	ROHTAK	XXXXXXXXXX	20
4	SURESH	Delhi	XXXXXXXXXX	18
3	SUJIT	ROHTAK	XXXXXXXXXX	20
2	RAMESH	GURGAON	XXXXXXXXXX	18
5	PRATIK	null	null	19

DML - SQL: INSERT INTO Statement

Inserting specific columns of a table: We can copy only those columns of a table which we want to insert into in a different table.

Syntax:



```
INSERT INTO first_table SELECT * FROM second_table;
```

first_table: *name of first table.*

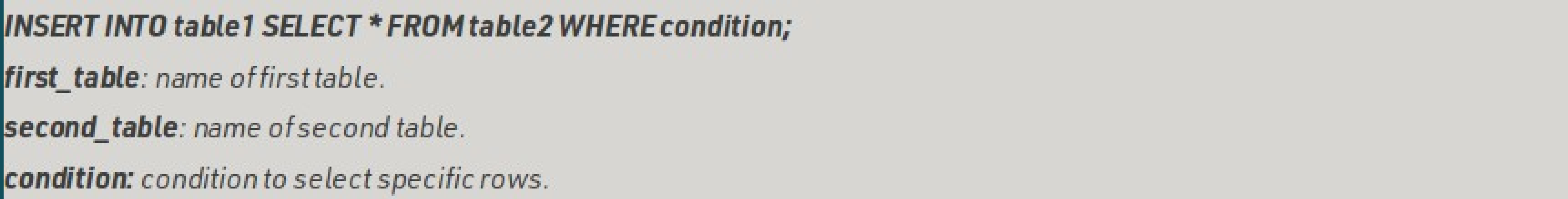
second_table: *name of second table.*

We have used the SELECT statement to copy the data of the selected columns only from the second table and INSERT INTO statement to insert in first table.

Copying specific rows from a table: We can copy specific rows from a table to insert into another table by using WHERE clause with the SELECT statement. We have to provide appropriate condition in the WHERE clause to select specific rows.

DML - SQL: INSERT INTO Statement

Copying specific rows from a table: We can copy specific rows from a table to insert into another table by using WHERE clause with the SELECT statement. We have to provide appropriate condition in the WHERE clause to select specific rows.



INSERT INTO table1 SELECT * FROM table2 WHERE condition;

***first_table**: name of first table.*

***second_table**: name of second table.*

***condition:** condition to select specific rows.*

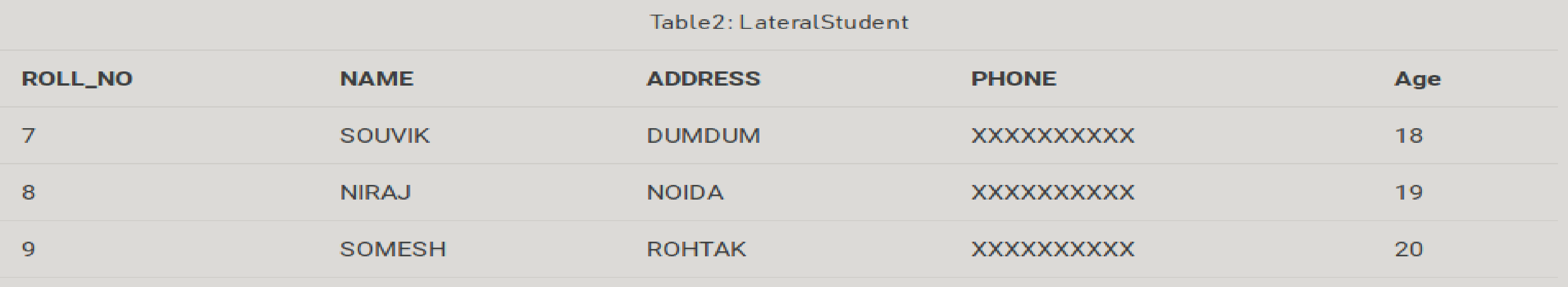


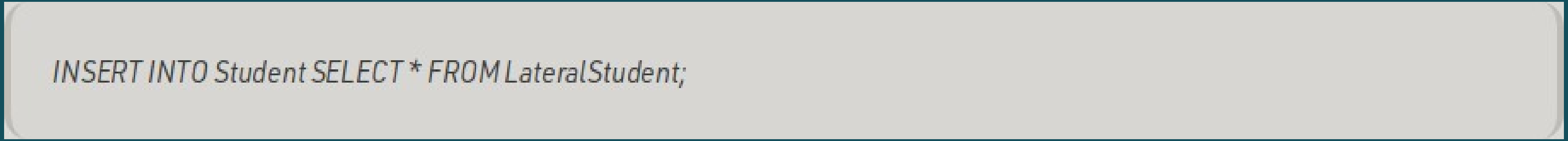
Table2: LateralStudent

ROLL_NO	NAME	ADDRESS	PHONE	Age
7	SOUVIK	DUMDUM	XXXXXXXXXX	18
8	NIRAJ	NOIDA	XXXXXXXXXX	19
9	SOMESH	ROHTAK	XXXXXXXXXX	20

DML - SQL: INSERT INTO Statement

Queries:

Method 1(Inserting all rows and columns):

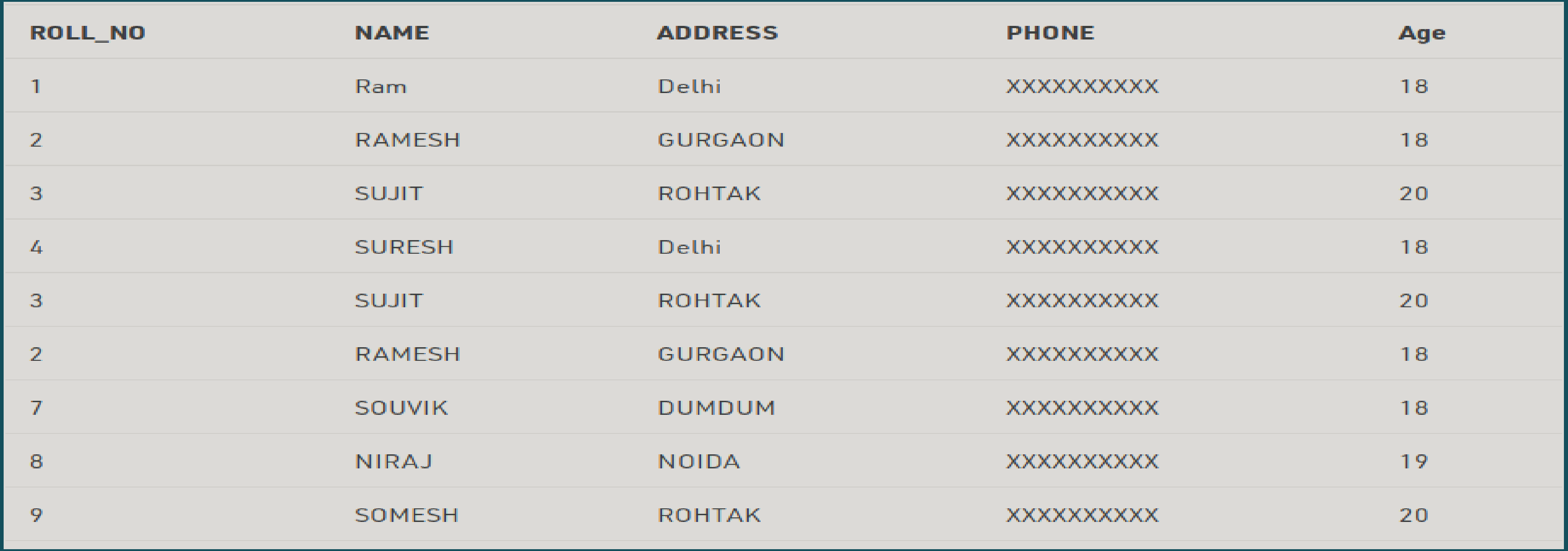


```
INSERT INTO Student SELECT * FROM LateralStudent;
```

DML - SQL: INSERT INTO Statement

Output:

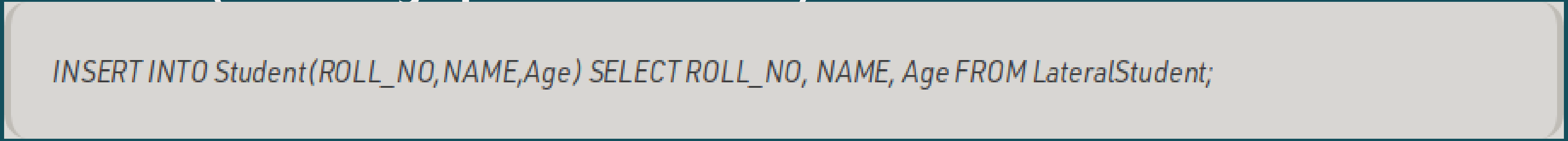
This query will insert all the data of the table LateralStudent in the table Student. The table Student will now look like,



ROLL_NO	NAME	ADDRESS	PHONE	Age
1	Ram	Delhi	XXXXXXXXXX	18
2	RAMESH	GURGAON	XXXXXXXXXX	18
3	SUJIT	ROHTAK	XXXXXXXXXX	20
4	SURESH	Delhi	XXXXXXXXXX	18
3	SUJIT	ROHTAK	XXXXXXXXXX	20
2	RAMESH	GURGAON	XXXXXXXXXX	18
7	SOUVIK	DUMDUM	XXXXXXXXXX	18
8	NIRAJ	NOIDA	XXXXXXXXXX	19
9	SOMESH	ROHTAK	XXXXXXXXXX	20

DML - SQL: INSERT INTO Statement

Method 2(Inserting specific columns):

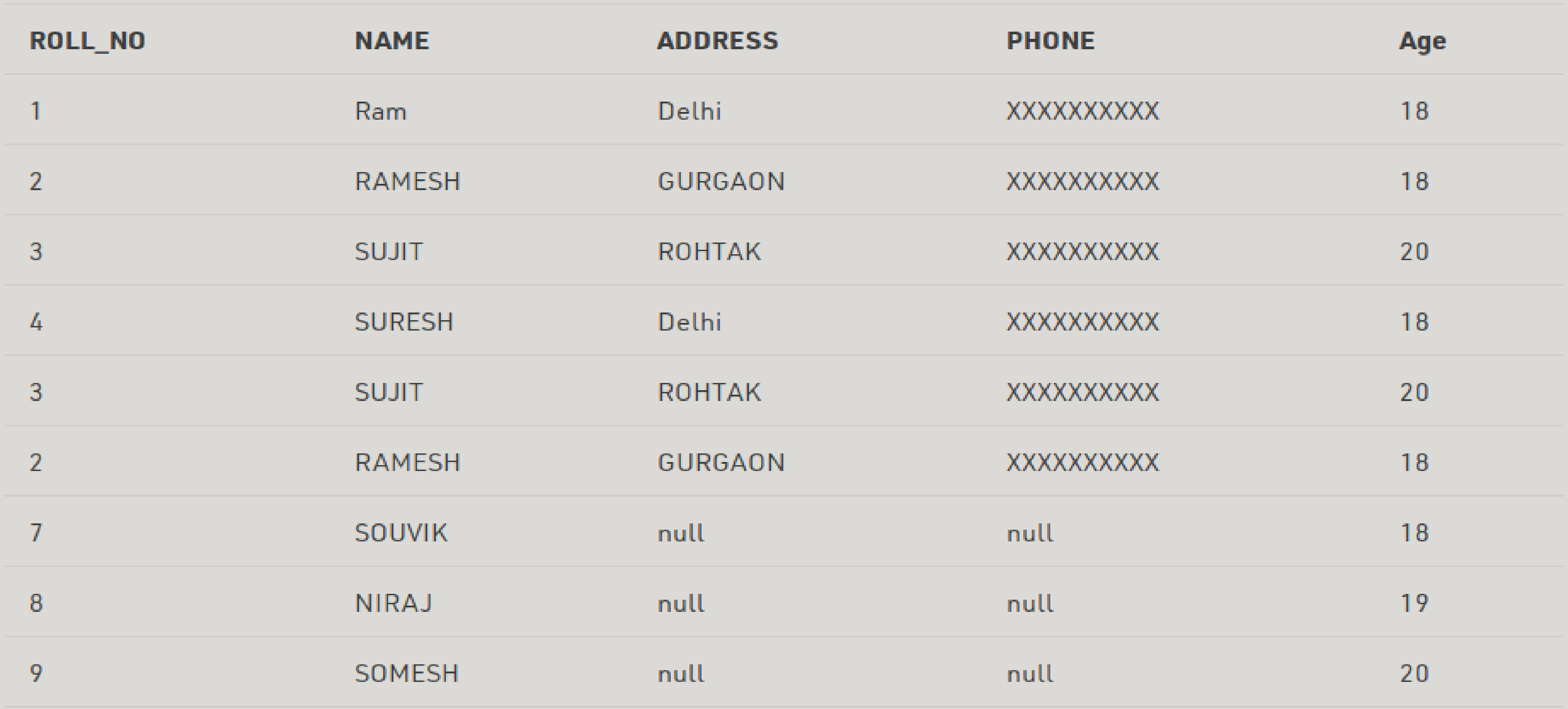


```
INSERT INTO Student(ROLL_NO,NAME,Age) SELECT ROLL_NO, NAME, Age FROM LateralStudent;
```

Output:

This query will insert the data in the columns ROLL_NO, NAME and Age of the table LateralStudent in the table Student and the remaining columns in the Student table will be filled by null which is the default value of the remaining columns. The table Student will now look like,

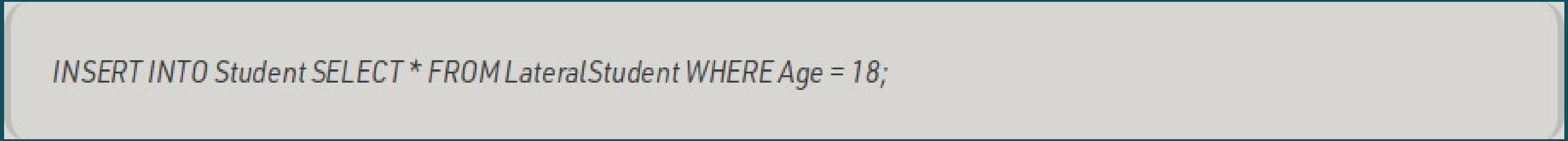
DML - SQL: INSERT INTO Statement



ROLL_NO	NAME	ADDRESS	PHONE	Age
1	Ram	Delhi	XXXXXXXXXX	18
2	RAMESH	GURGAON	XXXXXXXXXX	18
3	SUJIT	ROHTAK	XXXXXXXXXX	20
4	SURESH	Delhi	XXXXXXXXXX	18
3	SUJIT	ROHTAK	XXXXXXXXXX	20
2	RAMESH	GURGAON	XXXXXXXXXX	18
7	SOUVIK	null	null	18
8	NIRAJ	null	null	19
9	SOMESH	null	null	20

DML - SQL: INSERT INTO Statement

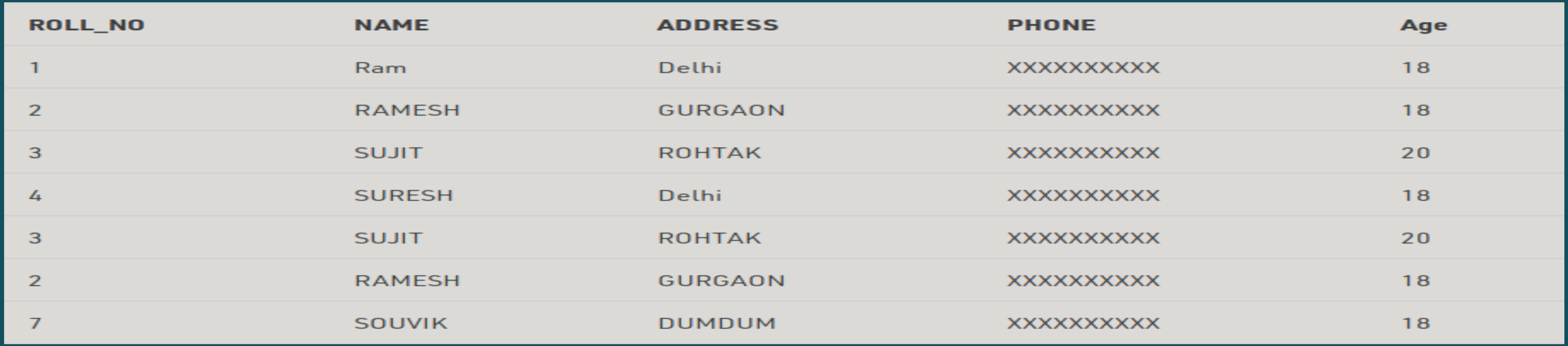
Select specific rows to insert:



```
INSERT INTO Student SELECT * FROM LateralStudent WHERE Age = 18;
```

Output:

This query will select only the first row from table LateralStudent to insert into the table Student. The table Student will now look like,



ROLL_NO	NAME	ADDRESS	PHONE	Age
1	Ram	Delhi	XXXXXXXXXX	18
2	RAMESH	GURGAON	XXXXXXXXXX	18
3	SUJIT	ROHTAK	XXXXXXXXXX	20
4	SURESH	Delhi	XXXXXXXXXX	18
3	SUJIT	ROHTAK	XXXXXXXXXX	20
2	RAMESH	GURGAON	XXXXXXXXXX	18
7	SOUVIK	DUMDUM	XXXXXXXXXX	18

DML - SQL: INSERT INTO Statement

To insert multiple rows in a table using Single SQL Statement

```
INSERT INTO table_name(Column1,Column2,Column3,.......)
VALUES (Value1, Value2,Value3,.....),
       (Value1, Value2,Value3,.....),
        (Value1, Value2,Value3,.....),
        ............................ ;
```

table_name: name of the table

Column1: name of first column, second column …

Value1, Value2, Value3 : value of first column, second column,… for each new row inserted

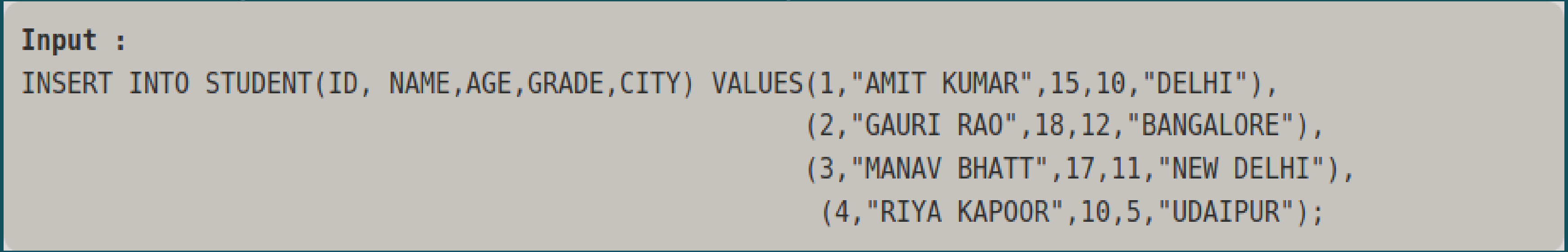
You need To provide Multiple lists of values where each list is separated by ",". Every list of value corresponds to values to be inserted in each new row of the table.

Values in the next list tells values to be inserted in the next Row of the table.

DML - SQL: INSERT INTO Statement

Example:

The following SQL statement insert multiple rows in Student Table.

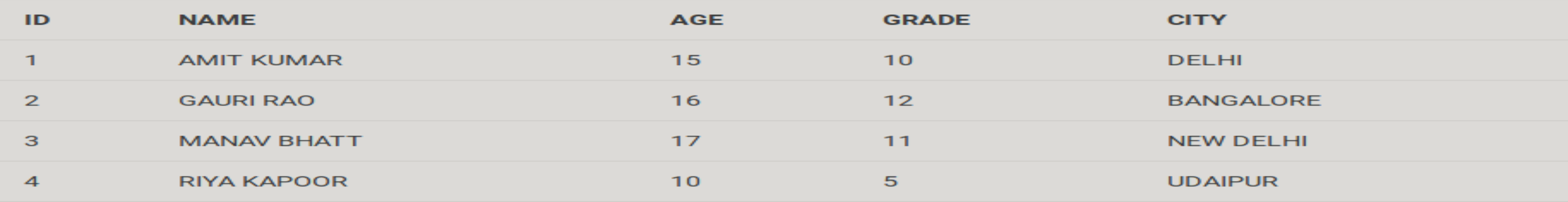


```
Input :
INSERT INTO STUDENT(ID, NAME,AGE,GRADE,CITY) VALUES(1,"AMIT KUMAR",15,10,"DELHI"),
                                                   (2,"GAURI RAO",18,12,"BANGALORE"),
                                                   (3,"MANAV BHATT",17,11,"NEW DELHI"),
                                                    (4,"RIYA KAPOOR",10,5,"UDAIPUR");
```

Output : STUDENT TABLE

This query will insert all values in each successive row in the STUDENT TABLE .

Thus STUDENT Table will look like this:

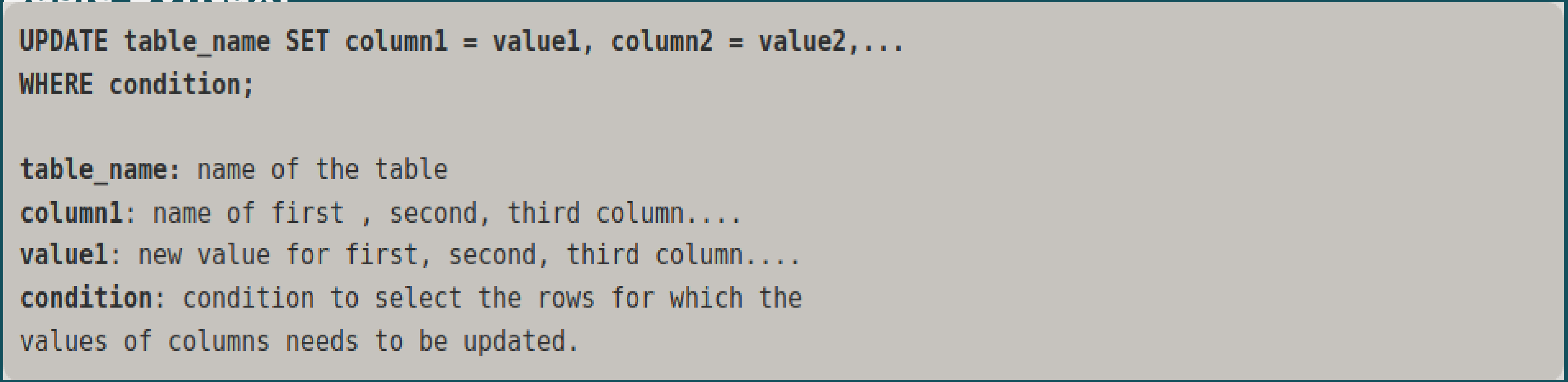


ID	NAME	AGE	GRADE	CITY
1	AMIT KUMAR	15	10	DELHI
2	GAURI RAO	16	12	BANGALORE
3	MANAV BHATT	17	11	NEW DELHI
4	RIYA KAPOOR	10	5	UDAIPUR

DML - SQL: UPDATE Statement

The UPDATE statement in SQL is used to update the data of an existing table in database. We can update single columns as well as multiple columns using UPDATE statement as per our requirement.

Basic Syntax:

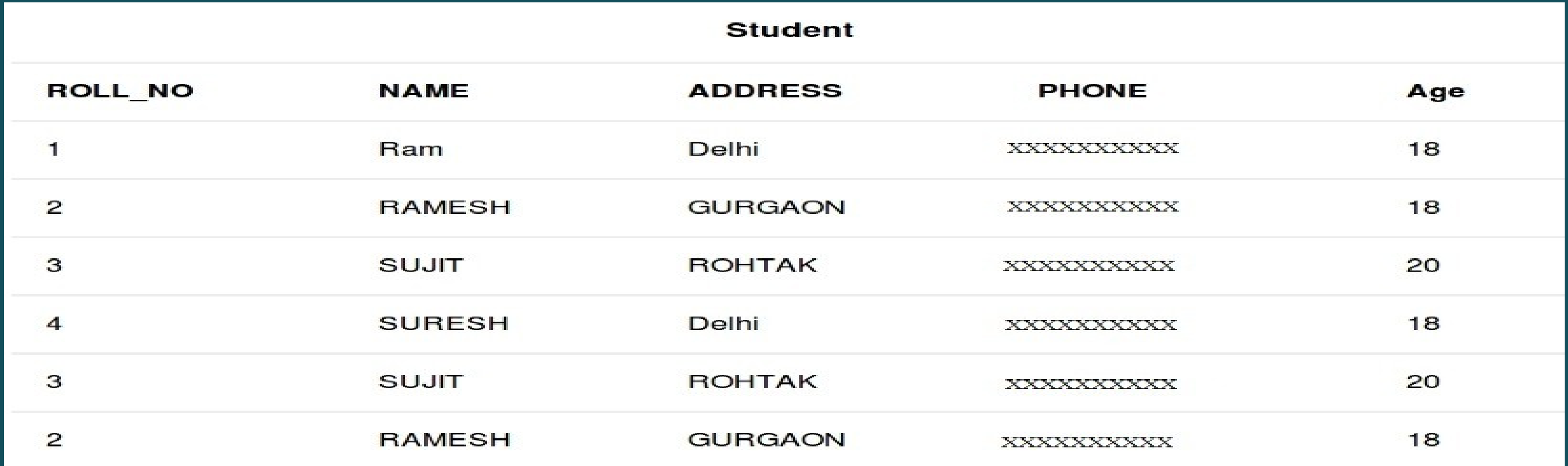


```
UPDATE table_name SET column1 = value1, column2 = value2,...
WHERE condition;

table_name: name of the table
column1: name of first , second, third column....
value1: new value for first, second, third column....
condition: condition to select the rows for which the
values of columns needs to be updated.
```

DML - SQL: UPDATE Statement

NOTE: In the above query the **SET** statement is used to set new values to the particular column and the **WHERE** clause is used to select the rows for which the columns are needed to be updated. If we have not used the WHERE clause then the columns in **all** the rows will be updated. So the WHERE clause is used to choose the particular rows.



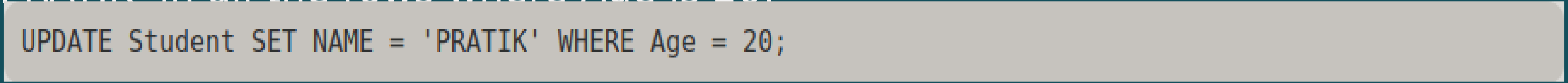
Student

ROLL_NO	NAME	ADDRESS	PHONE	Age
1	Ram	Delhi	XXXXXXXXXX	18
2	RAMESH	GURGAON	XXXXXXXXXX	18
3	SUJIT	ROHTAK	XXXXXXXXXX	20
4	SURESH	Delhi	XXXXXXXXXX	18
3	SUJIT	ROHTAK	XXXXXXXXXX	20
2	RAMESH	GURGAON	XXXXXXXXXX	18

DML - SQL: UPDATE Statement

Example Queries:

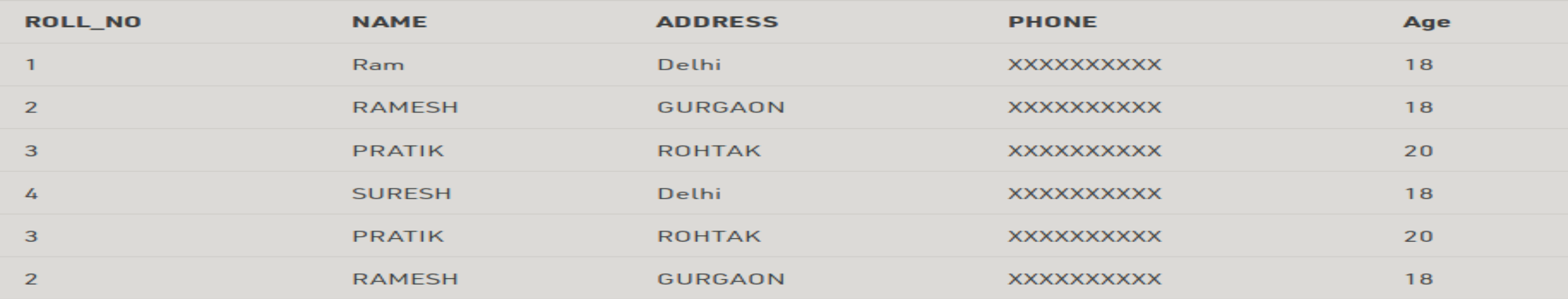
Updating single column: Update the column NAME and set the value to 'PRATIK' in all the rows where Age is 20.



```
UPDATE Student SET NAME = 'PRATIK' WHERE Age = 20;
```

Output:

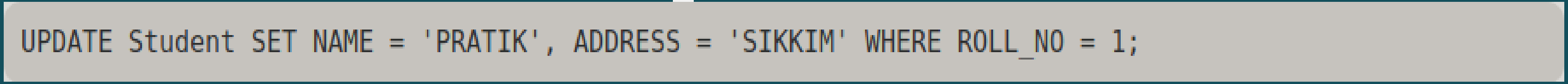
This query will update two rows(third row and fifth row) and the table Student will now look like,



ROLL_NO	NAME	ADDRESS	PHONE	Age
1	Ram	Delhi	XXXXXXXXXX	18
2	RAMESH	GURGAON	XXXXXXXXXX	18
3	PRATIK	ROHTAK	XXXXXXXXXX	20
4	SURESH	Delhi	XXXXXXXXXX	18
3	PRATIK	ROHTAK	XXXXXXXXXX	20
2	RAMESH	GURGAON	XXXXXXXXXX	18

DML - SQL: UPDATE Statement

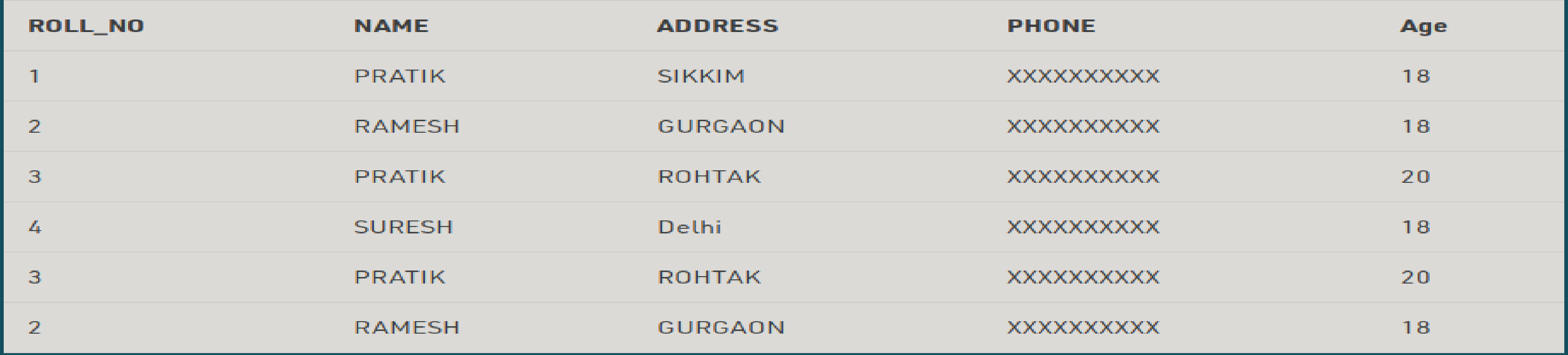
Updating multiple columns: Update the columns NAME to ‘PRATIK’ and ADDRESS to ‘SIKKIM’ where ROLL_NO is 1.



```
UPDATE Student SET NAME = 'PRATIK', ADDRESS = 'SIKKIM' WHERE ROLL_NO = 1;
```

Output:

The above query will update two columns in the first row and the table **Student** will now look like,

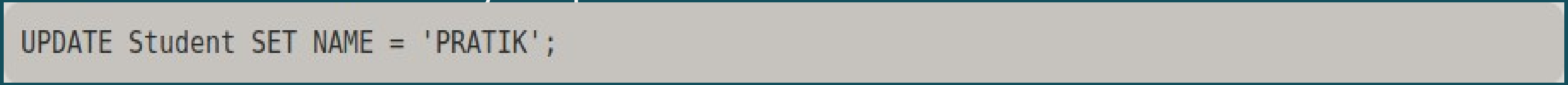


ROLL_NO	NAME	ADDRESS	PHONE	Age
1	PRATIK	SIKKIM	XXXXXXXXXX	18
2	RAMESH	GURGAON	XXXXXXXXXX	18
3	PRATIK	ROHTAK	XXXXXXXXXX	20
4	SURESH	Delhi	XXXXXXXXXX	18
3	PRATIK	ROHTAK	XXXXXXXXXX	20
2	RAMESH	GURGAON	XXXXXXXXXX	18

DML - SQL: UPDATE Statement

Note: For updating multiple columns we have used comma(,) to separate the names and values of two columns.

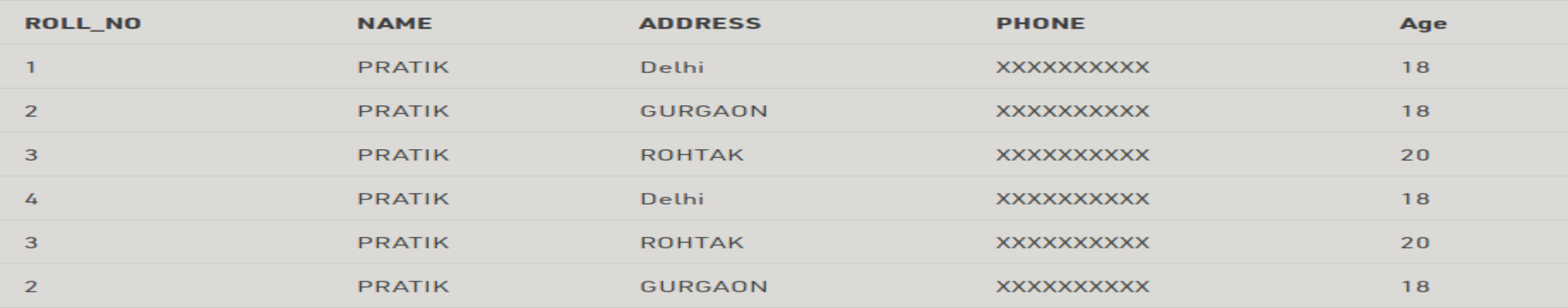
Omitting WHERE clause: If we omit the WHERE clause from the update query then all of the rows will get updated.



```
UPDATE Student SET NAME = 'PRATIK';
```

Output:

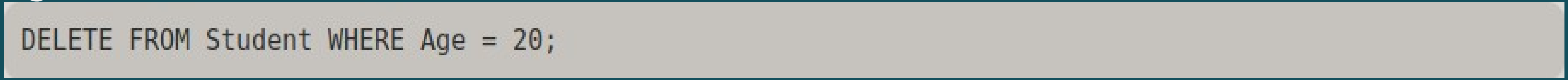
The table **Student** will now look like,



ROLL_NO	NAME	ADDRESS	PHONE	Age
1	PRATIK	Delhi	XXXXXXXXXX	18
2	PRATIK	GURGAON	XXXXXXXXXX	18
3	PRATIK	ROHTAK	XXXXXXXXXX	20
4	PRATIK	Delhi	XXXXXXXXXX	18
3	PRATIK	ROHTAK	XXXXXXXXXX	20
2	PRATIK	GURGAON	XXXXXXXXXX	18

DML - SQL: DELETE Statement

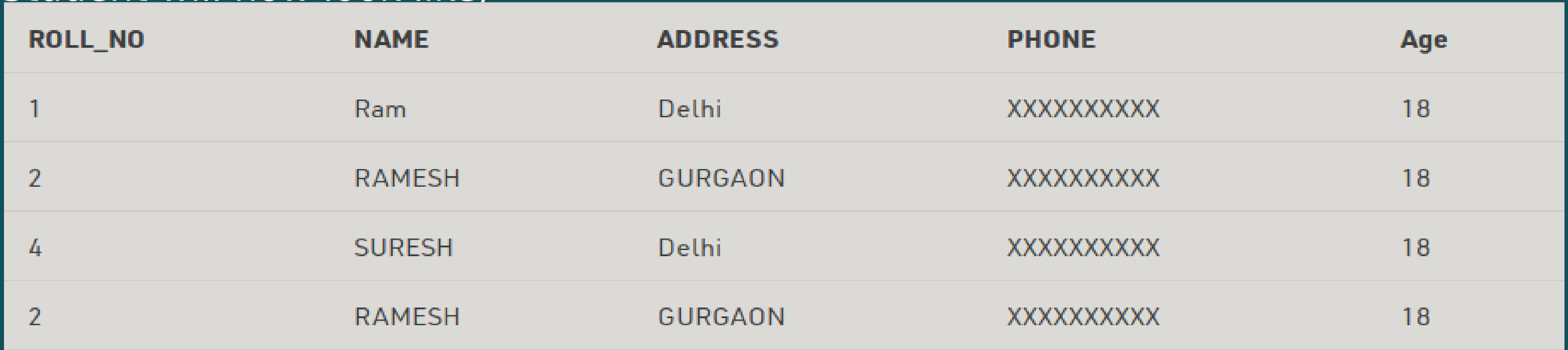
Deleting multiple records: Delete the rows from the table Student where Age is 20. This will delete 2 rows(third row and fifth row).



```
DELETE FROM Student WHERE Age = 20;
```

Output:

The above query will delete two rows(third row and fifth row) and the table Student will now look like,

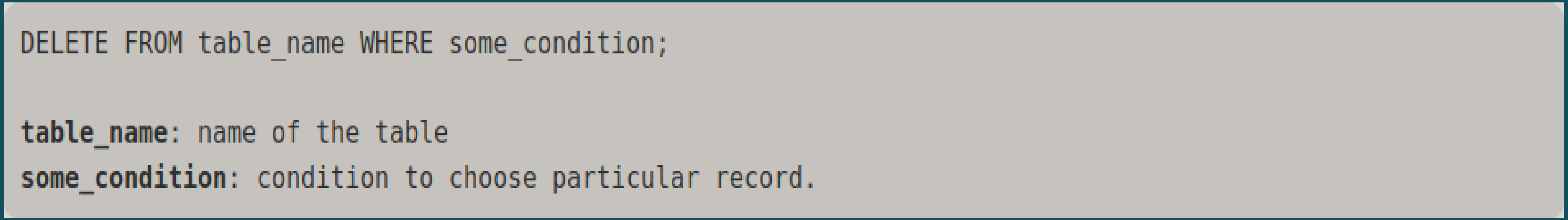


ROLL_NO	NAME	ADDRESS	PHONE	Age
1	Ram	Delhi	XXXXXXXXXX	18
2	RAMESH	GURGAON	XXXXXXXXXX	18
4	SURESH	Delhi	XXXXXXXXXX	18
2	RAMESH	GURGAON	XXXXXXXXXX	18

DML - SQL: DELETE Statement

The DELETE Statement in SQL is used to delete existing records from a table. We can delete a single record or multiple records depending on the condition we specify in the WHERE clause.

Basic Syntax:



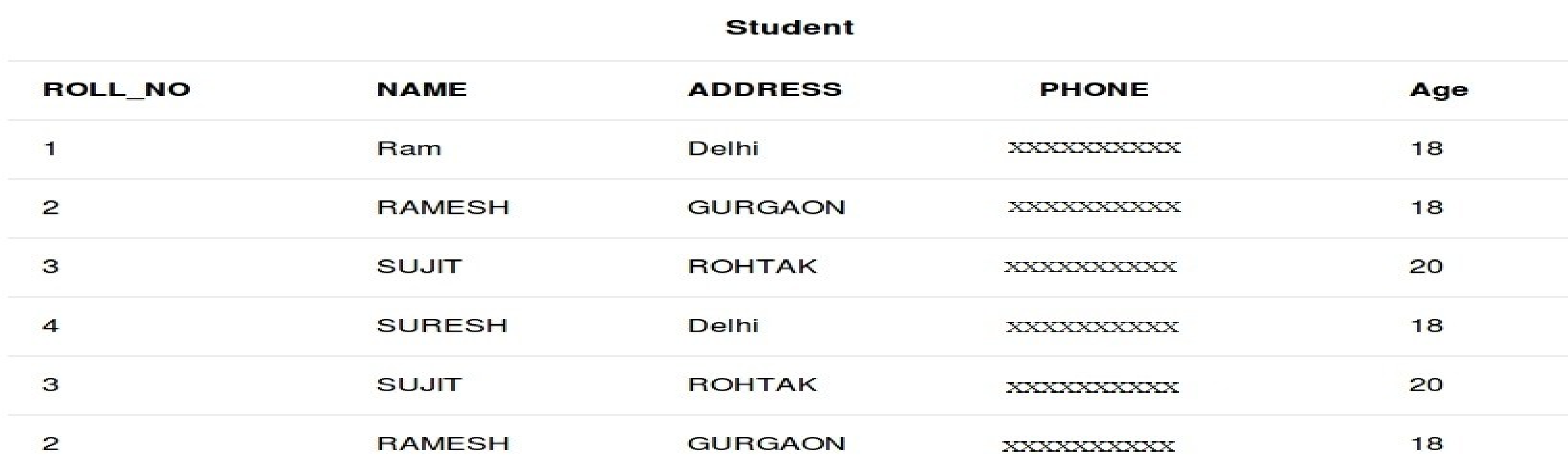
```
DELETE FROM table_name WHERE some_condition;

table_name: name of the table
some_condition: condition to choose particular record.
```

DML - SQL: DELETE Statement

Note: We can delete single as well as multiple records depending on the condition we provide in WHERE clause. If we omit the WHERE clause then all of the records will be deleted and the table will be empty.

Sample Table:



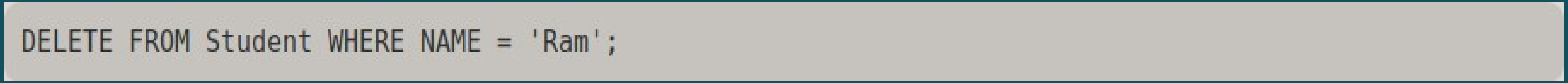
Student

ROLL_NO	NAME	ADDRESS	PHONE	Age
1	Ram	Delhi	XXXXXXXXXX	18
2	RAMESH	GURGAON	XXXXXXXXXX	18
3	SUJIT	ROHTAK	XXXXXXXXXX	20
4	SURESH	Delhi	XXXXXXXXXX	18
3	SUJIT	ROHTAK	XXXXXXXXXX	20
2	RAMESH	GURGAON	XXXXXXXXXX	18

DML - SQL: DELETE Statement

Example Queries:

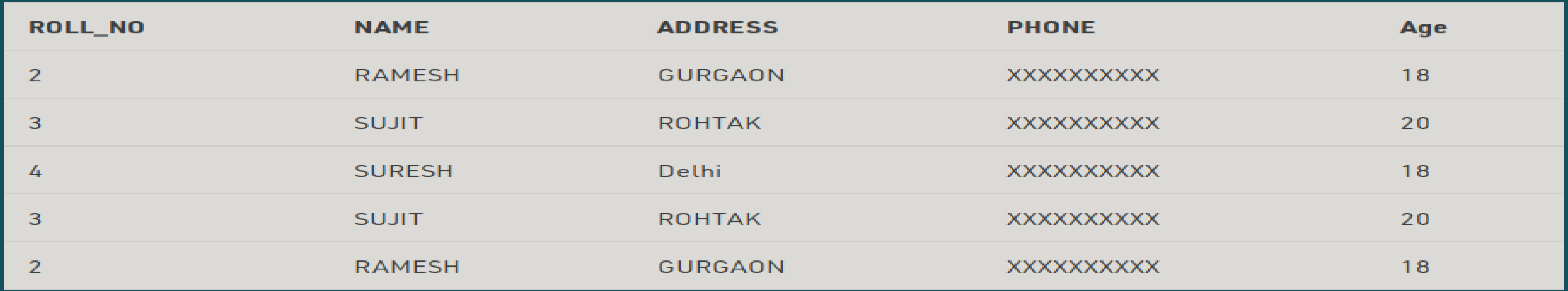
Deleting single record: Delete the rows where NAME = ‘Ram’. This will delete only the first row.



```
DELETE FROM Student WHERE NAME = 'Ram';
```

Output:

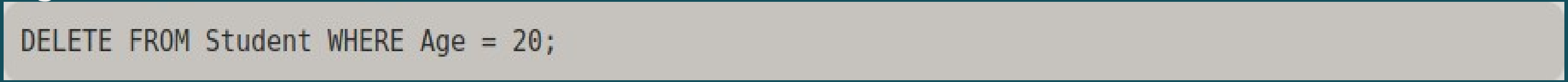
The above query will delete only the first row and the table Student will now look like,



ROLL_NO	NAME	ADDRESS	PHONE	Age
2	RAMESH	GURGAON	XXXXXXXXXX	18
3	SUJIT	ROHTAK	XXXXXXXXXX	20
4	SURESH	Delhi	XXXXXXXXXX	18
3	SUJIT	ROHTAK	XXXXXXXXXX	20
2	RAMESH	GURGAON	XXXXXXXXXX	18

DML - SQL: DELETE Statement

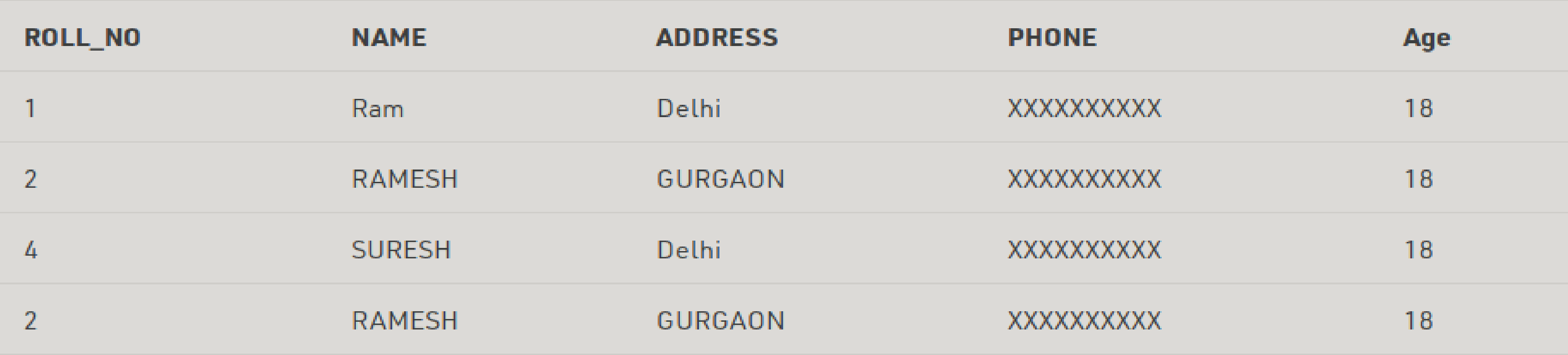
Deleting multiple records: Delete the rows from the table Student where Age is 20. This will delete 2 rows(third row and fifth row).



```
DELETE FROM Student WHERE Age = 20;
```

Output:

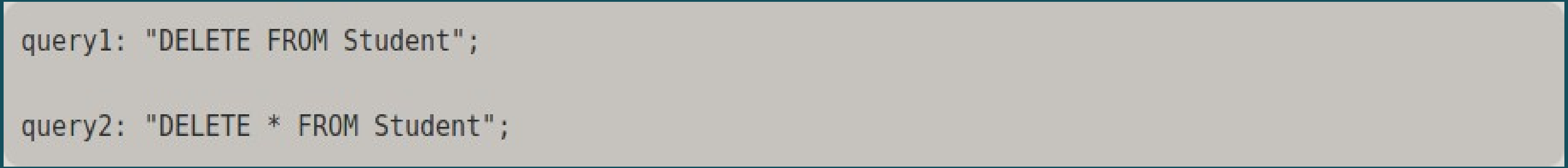
The above query will delete two rows(third row and fifth row) and the table Student will now look like,



ROLL_NO	NAME	ADDRESS	PHONE	Age
1	Ram	Delhi	XXXXXXXXXX	18
2	RAMESH	GURGAON	XXXXXXXXXX	18
4	SURESH	Delhi	XXXXXXXXXX	18
2	RAMESH	GURGAON	XXXXXXXXXX	18

DML - SQL: DELETE Statement

Delete all of the records: There are two queries to do this as shown below,



```
query1: "DELETE FROM Student";

query2: "DELETE * FROM Student";
```

Output:

All of the records in the table will be deleted, there are no records left to display. The table Student will become empty!

Congratulations!