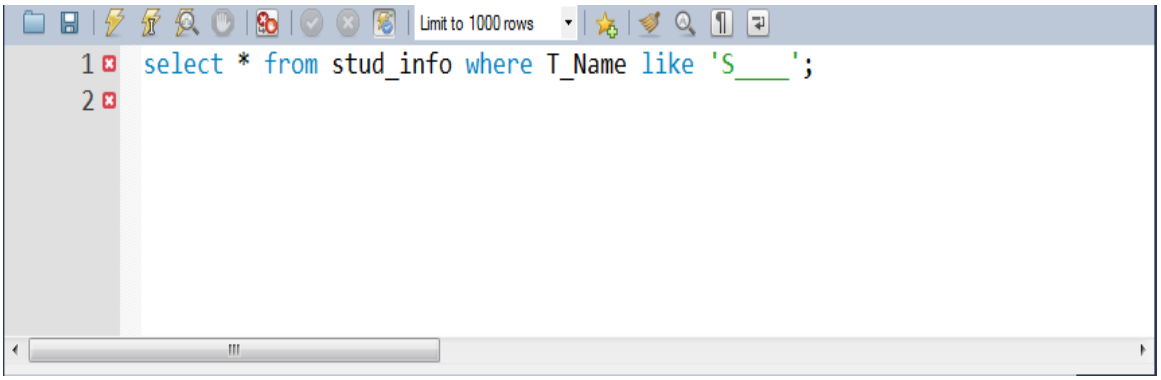
Contents

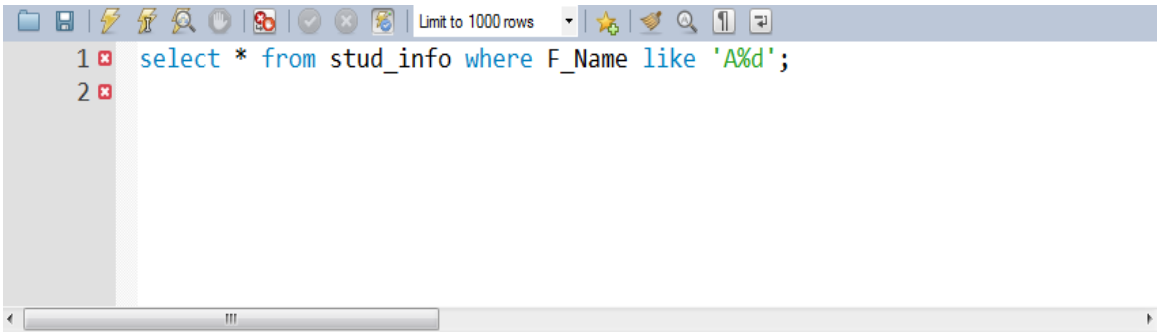
1. Examples using LIKE

Statement	Description
WHERE SALARY LIKE '200%'	Finds any values that start with 200
WHERE SALARY LIKE '%200%'	Finds any values that have 200 in any position
WHERE SALARY LIKE '_00%'	Finds any values that have 00 in the second and third positions
WHERE SALARY LIKE '2_%_%'	Finds any values that start with 2 and are at least 3 characters in length
WHERE SALARY LIKE '%2'	Finds any values that end with 2
WHERE SALARY LIKE '_2%3'	Finds any values that have a 2 in the second position and end with a 3
WHERE SALARY LIKE '2___3'	Finds any values in a five-digit number that start with 2 and end with 3

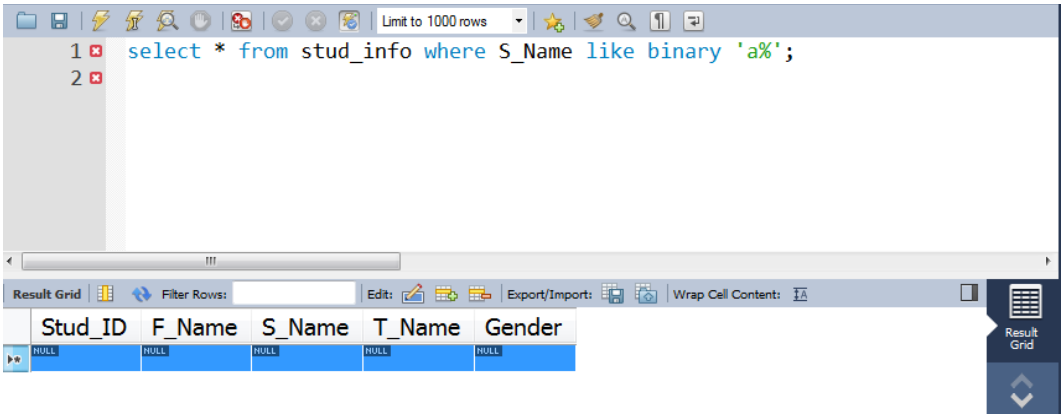
Example: find the tuples which third names start with ‘S’ and consist of five characters:



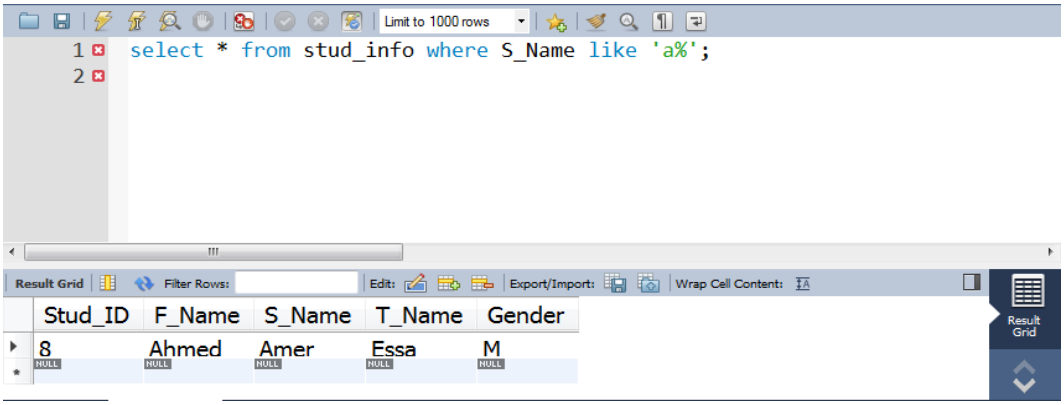
Example: find the tuples which first name start with ‘A’ and end with ‘d’:



-Like clause searches for non-case sensitive characters, so if you want to search for case sensitive character use function BINARY:

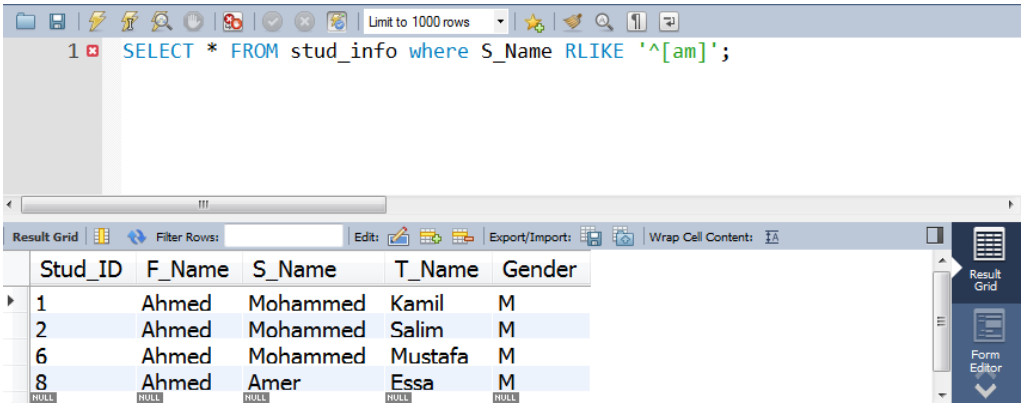


The result without binary:

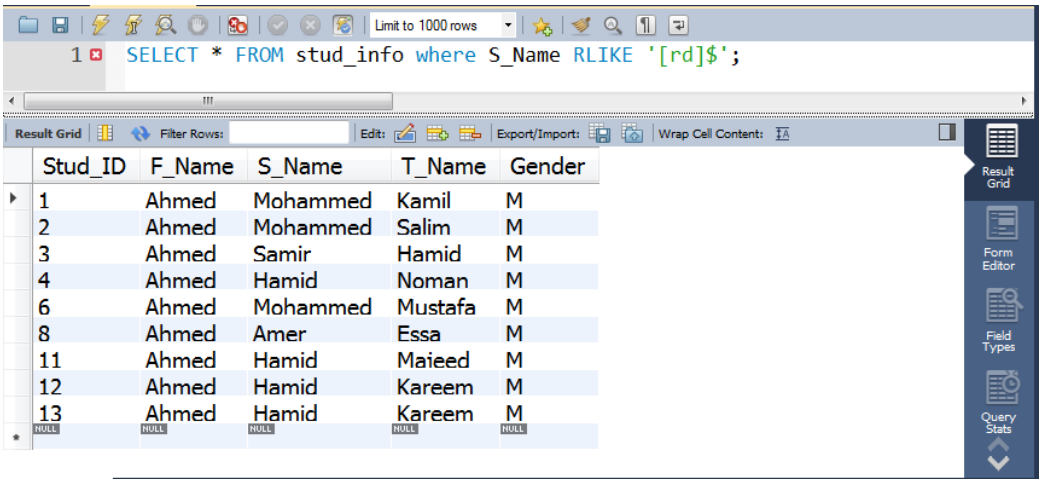


-RLIKE clause can be used for searching specific characters in the start or the end of string. Use the character '^' to refer to the start character(s) in the first of the string, or use '$' to search the ending character(s). To specify set of characters use [charlist] after RLIKE.

Example: find the tuples which S_Name start with characters ’a’ or ‘m’:



Example: find the tuples which S_Name ending with characters ’d’ or ‘r’:



Examples:

RLIKE '[k-z]$'	ending with characters ‘k’ to ‘z’
RLIKE '^[a-n]'	starting with characters ‘a’ to ‘n’
RLIKE '^id$'	starting and ending with’ id’
NOT RLIKE '^[a]'	not starting with ‘a’ character
NOT RLIKE 'id$'	not ending with ‘id’

2. DML Data Manipulation Language

Command	Description
INSERT	Creates a record
UPDATE	Modifies records
DELETE	Deletes records

3. Insert Statement

The SQL *INSERT INTO* Statement is used to add new rows of data to a table in the database.

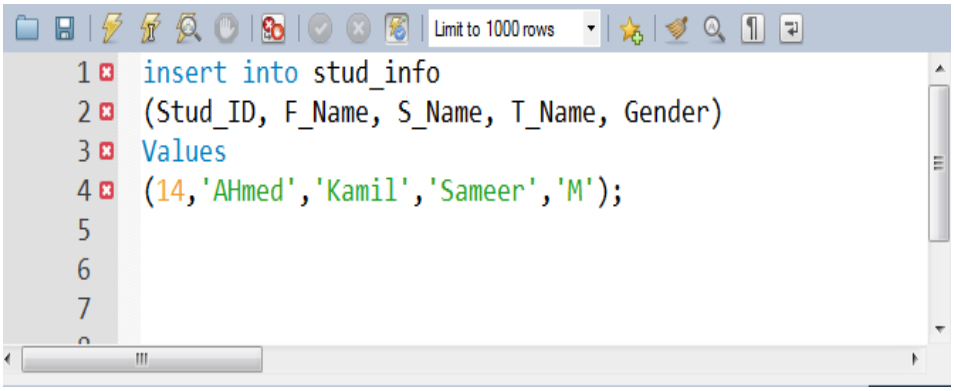
```
INSERT INTO TABLE_NAME (column1, column2, column3,...columnN)]
VALUES (value1, value2, value3,...valueN);
```

Here, column1, column2,...columnN are the names of the columns in the table into which you want to insert data.

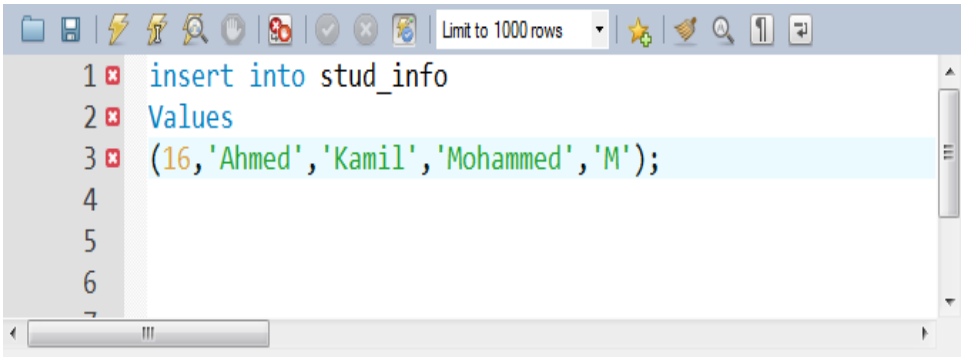
You may not need to specify the column(s) name in the SQL query if you are adding values for all the columns of the table. But make sure the order of the values is in the same order as the columns in the table. The SQL INSERT INTO syntax would be as follows:

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

Example: Insert new record to table stud_info with attributes (Stud_ID, F_Name, S_Name, T_Name, Gender).



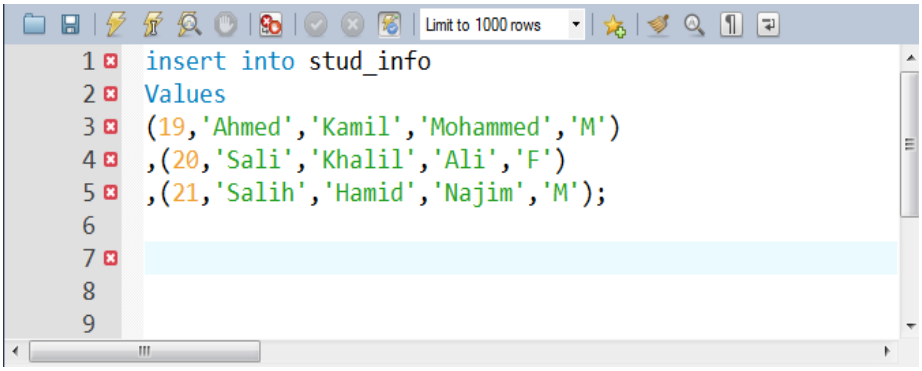
-To insert values without selecting columns remove columns after selecting table name but you should add all the values after keyword values in insert statement.



-To insert multiple values at the same time:

```
INSERT INTO TableName
(Column1, Column2, Column3, Column4)
Values
(Value1,Value2,Value3,Value4),
(Value5,Value6,Value7,Value8);
```

Example: Insert three records at the same time:

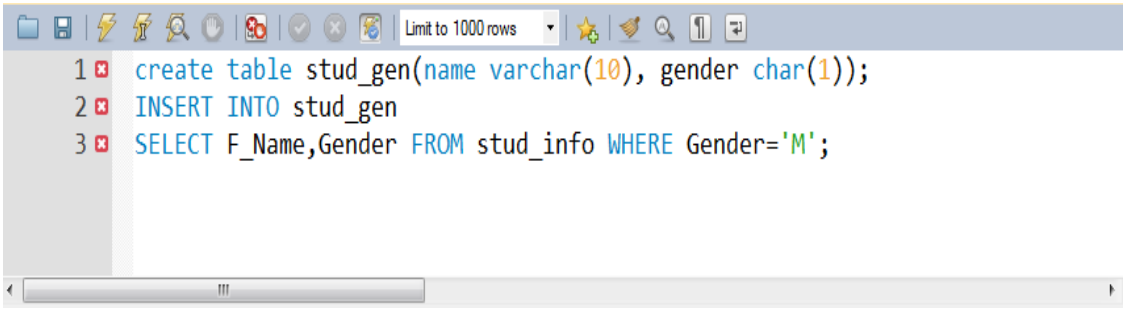


-Populate one table using another table:

You can populate data into a table through select statement over another table provided another table has a set of fields, which are required to populate first table. Here is the syntax:

```
INSERT INTO first_table_name [(column1, column2, ... columnN)]
   SELECT column1, column2, ...columnN
   FROM second_table_name
   [WHERE condition];
```

Example: based on the previous table, create new table with attributes (f_name,gender) and insert all tuples with Male gender:

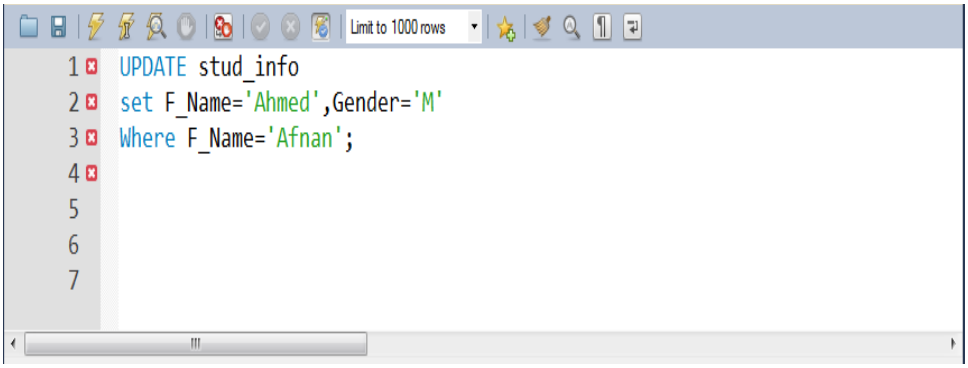


4. Update Statement

The SQL UPDATE Query is used to modify the existing records in a table. You can use WHERE clause with UPDATE query to update selected rows, otherwise all the rows would be affected.

```
UPDATE table_name
SET column1 = value1, column2 = value2...., columnN = valueN
WHERE [condition];
```

Example: update the tuple attributes F_Name and Gender and set F_Name=’Ahmed’ and Gender=’M’ for the record with F_Name=’Afnan’:



- If you remove the condition, the update will affect all the record, so be careful.

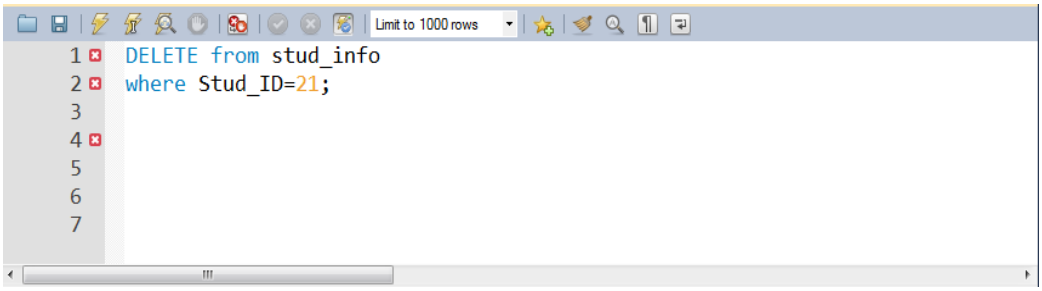
5. Delete Statement

The SQL **DELETE** Query is used to delete the existing records from a table. You can use WHERE clause with DELETE query to delete selected rows, otherwise all the records would be deleted.

```
DELETE FROM table_name
WHERE [condition];
```

-If you set delete without condition, you will delete all the records, so be careful.

Example: based on the previous table, delete the record with Stud_ID=21:



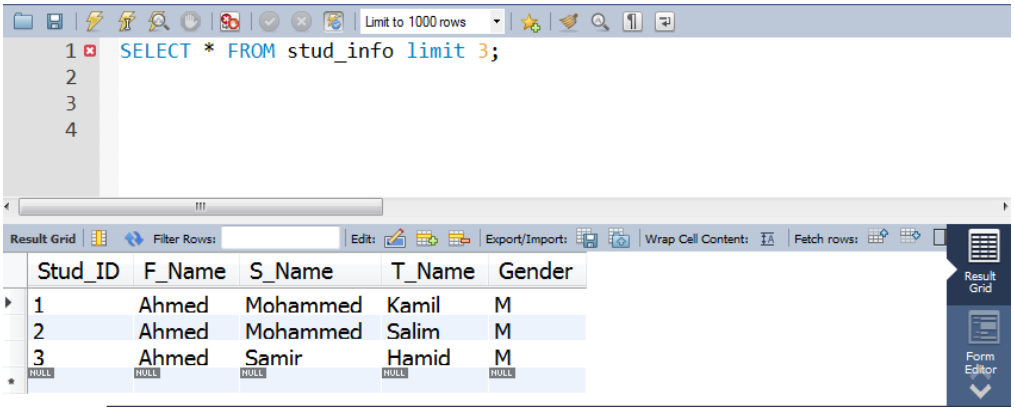
6. Limit Clause

The SQL **TOP** clause is used to fetch a TOP N number or X percent records from a table.

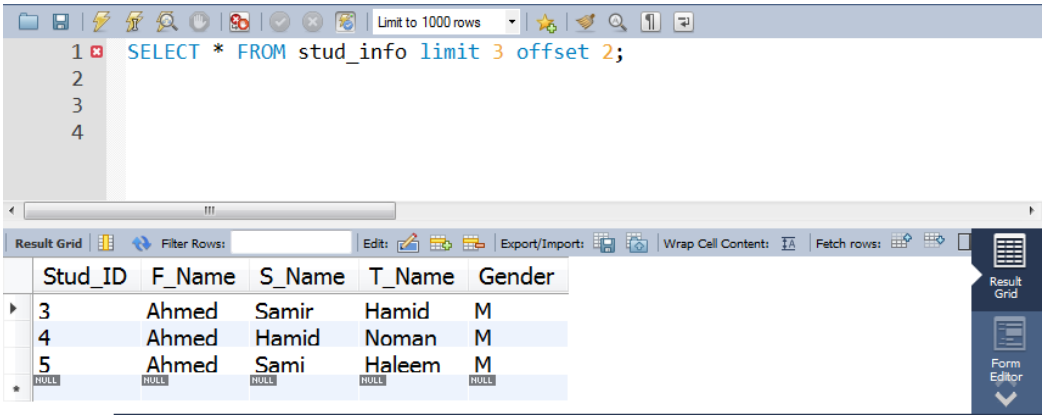
Note: All the databases do not support TOP clause. For example MySQL supports **LIMIT** clause to fetch limited number of records and Oracle uses **ROWNUM** to fetch limited number of records.

```
SELECT * FROM TableName limit NumberofRecords;
```

Example: Show only first three records from stud_info table:

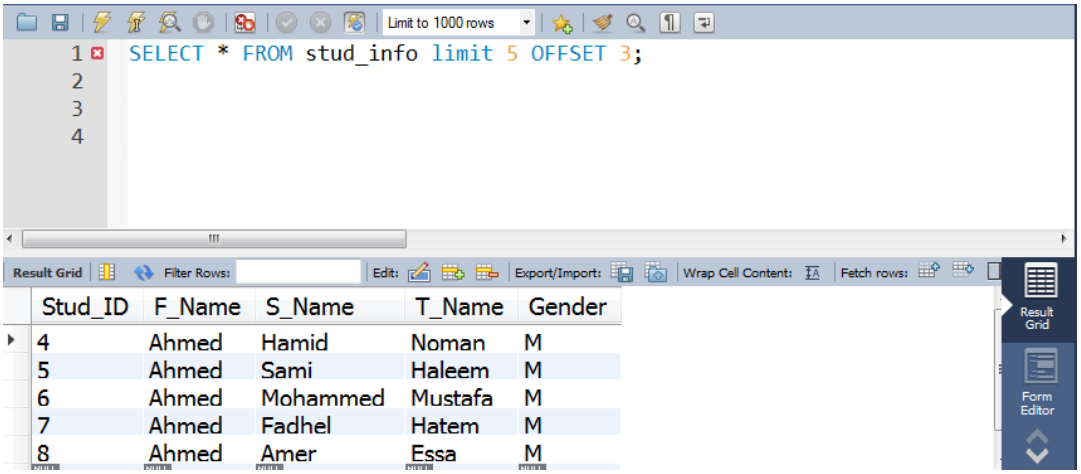


-You can define the OFFSET to select specific records, so if you want to select three records started from record 3:



-The started record took the value of OFFSET+1.

Example: show five records starting from record 4:



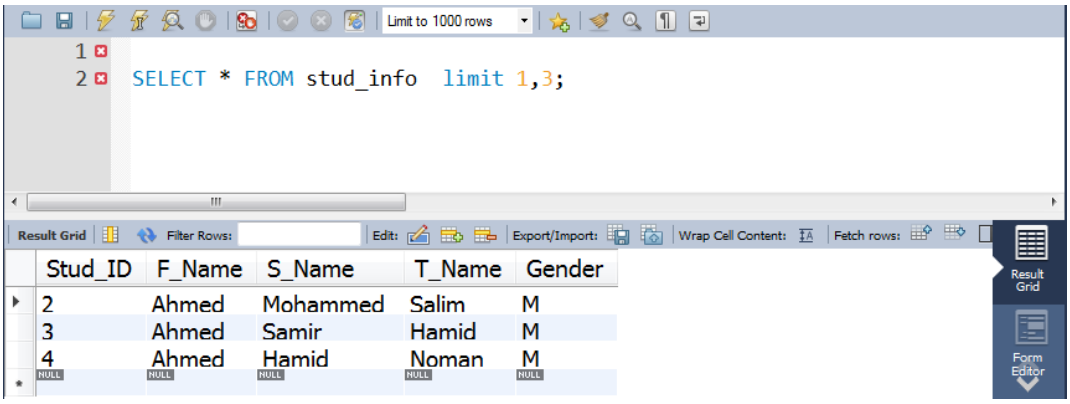
```
SELECT * FROM stud_info limit 5 OFFSET 3;
```

Stud_ID	F_Name	S_Name	T_Name	Gender
4	Ahmed	Hamid	Noman	M
5	Ahmed	Sami	Haleem	M
6	Ahmed	Mohammed	Mustafa	M
7	Ahmed	Fadhel	Hatem	M
8	Ahmed	Amer	Essa	M

-You can combine both LIMIT and OFFSET by using LIMT clause:

```
SELECT * FROM TableName LIMIT OFFSetValue, NumberOfRecords;
```

Example: show second three tuples from stud_info:



```
SELECT * FROM stud_info  limit 1,3;
```

Stud_ID	F_Name	S_Name	T_Name	Gender
2	Ahmed	Mohammed	Salim	M
3	Ahmed	Samir	Hamid	M
4	Ahmed	Hamid	Noman	M