

SQL CODE

CREATE TABLE

The screenshot shows a SQL File 1* window with the following code:

```
1 #CREATE DATABASE SIVA_8079893
2
3 • USE SIVA_8079893;
4 • CREATE TABLE STU_DENTS(
5   id INT NOT NULL AUTO_INCREMENT,
6   FirstName VARCHAR(20) NOT NULL,
7   LastName VARCHAR(20) NOT NULL,
8   Address VARCHAR(20) NOT NULL,
9   PRIMARY KEY (id)
10 );
11
```

The Output window shows the following results:

#	Time	Action	Message	Duration / Fetch
16	13:19:43	CREATE TABLE STUDENTS(id INT NOT NULL AUTO_INCREMENT, Name VARCHAR(20) NOT NULL, PRIMARY KEY (id))	Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'Name VARCHAR(20) NOT NULL, PRIMARY KEY (id))' at line 1	0.000 sec
17	13:23:06	USE SIVA_8079893	0 row(s) affected	0.000 sec
18	13:23:06	CREATE TABLE STU_DENTS(id INT NOT NULL AUTO_INCREMENT, FirstName VARCHAR(20) NOT NULL, LastName VARCHAR(20) NOT NULL, Address VARCHAR(20) NOT NULL, PRIMARY KEY (id))	Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'FirstName VARCHAR(20) NOT NULL, LastName VARCHAR(20) NOT NULL, Address VARCHAR(20) NOT NULL, PRIMARY KEY (id))' at line 1	0.000 sec
19	13:23:18	USE SIVA_8079893	0 row(s) affected	0.000 sec
20	13:23:18	CREATE TABLE STU_DENTS(id INT NOT NULL AUTO_INCREMENT, FirstName VARCHAR(20) NOT NULL, LastName VARCHAR(20) NOT NULL, Address VARCHAR(20) NOT NULL, PRIMARY KEY (id))	0 row(s) affected	0.672 sec
21	13:23:32	USE SIVA_8079893	0 row(s) affected	0.000 sec

INSERT TABLE

The screenshot shows a SQL File 1* window with the following code:

```
12 • INSERT INTO STU_DENTS( FirstName,LastName,Address)
13   VALUES("ABC", "b", "Chennai");
14 • INSERT INTO STU_DENTS( FirstName,LastName,Address)
15   VALUES("Nivetha", "R", "Hydrabad");
16 • INSERT INTO STU_DENTS( FirstName,LastName,Address)
17   VALUES("Priya", "C", "Lia");
18 • SELECT * FROM STU DENTS;
```

The Result Grid shows the following data:

	id	FirstName	LastName	Address
1	1	ABC	b	Chennai
2	2	Nivetha	R	Hydrabad
3	3	Priya	C	Lia
4	NULL	NULL	NULL	NULL

Table 2

SQL File 1* SQL File 3* x

Limit to 1000 rows

```

1 • INSERT INTO FRIENDS (id,CharacterName,OriginalName,Age) VALUES (1, "Monica Geller", "Courteney Cox", 52);
2 • INSERT INTO FRIENDS (id,CharacterName,OriginalName,Age) VALUES (2, "Rachel Green", "Jennifer Aniston",50);
3 • INSERT INTO FRIENDS (id,CharacterName,OriginalName,Age) VALUES (3, "Pheobe Buffay", "Lisa Kudrow", 58);
4 • INSERT INTO FRIENDS (id,CharacterName,OriginalName,Age) VALUES (4, "Chandler Bing", "Mathew Perry", 56);
5 • INSERT INTO FRIENDS (id,CharacterName,OriginalName,Age) VALUES (5, "Joey Tribbiani", "Matt Leblanc", 52);
6 • INSERT INTO FRIENDS (id,CharacterName,OriginalName,Age) VALUES (6, "Ross Geller", "David Schwimmer", 55);
7
8 • select * FROM FRIENDS;

```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: |

	id	CharacterName	OriginalNAME	Age
▶	1	Monica Geller	Courteney Cox	52
	2	Rachel Green	Jennifer Aniston	50
	3	Pheobe Buffay	Lisa Kudrow	58
	4	Chandler Bing	Mathew Perry	56
	5	Joey Tribbiani	Matt Leblanc	52
	6	Ross Geller	David Schwimmer	55
*	NULL	NULL	NULL	NULL

FRIENDS 1 x Apply Revert

UPDATE DATA

```

4
5 • Update FRIENDS SET OriginalName = "David Schwimmer"
6   WHERE id = 7;
7
8 • Select * FROM FRIENDS;

```

Result Grid | Filter Rows: | Edit: | Export/Import: |

	id	CharacterName	OriginalNAME	Age
▶	1	Monica Geller	Courteney Cox	52
	2	Rachel Green	Jennifer Aniston	50
	3	Pheobe Buffay	Lisa Kudrow	58
	4	Chandler Bing	Mathew Perry	56
	5	Joey Tribbiani	Matt Leblanc	52
	6	Ross Geller	David Schwimmer	55
	7	Ross Geller	David Schwimmer	56
*	NULL	NULL	NULL	NULL

DELETE DATA

```
8
9 • DELETE FROM FRIENDS WHERE id = 7;
10 • Select * FROM FRIENDS;
```

Result Grid | Filter Rows: | Edit:

	id	CharacterName	OriginalNAME	Age
▶	1	Monica Geller	Courteney Cox	52
	2	Rachel Green	Jennifer Aniston	50
	3	Pheobe Buffay	Lisa Kudrow	58
	4	Chandler Bing	Mathew Perry	56
	5	Joey Tribbiani	Matt Leblanc	52
	6	Ross Geller	David Schwimmer	55
•	NULL	NULL	NULL	NULL

LIMIT

SQL File 1* SQL File 3* SQL File 4* SQL File 5* SQL File 6* SQL File 7* x

```
1 • SELECT * FROM FRIENDS
2 LIMIT 3;
```

Limit to 1000 rows

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | Fetch rows:

	id	CharacterName	OriginalNAME	Age
▶	1	Monica Geller	Courteney Cox	52
	2	Rachel Green	Jennifer Aniston	50
	3	Pheobe Buffay	Lisa Kudrow	58
•	NULL	NULL	NULL	NULL

Result Grid
Form Editor

WHERE CLAUSE

```
19 • SELECT * FROM STU_DENTS where address LIKE "c%";
```

Result Grid | Filter Rows: | Edit: | Exp

	id	FirstName	LastNAME	Address
▶	1	ABC	b	Chennai
•	NULL	NULL	NULL	NULL

18 • `SELECT * FROM STU_DENTS where id = 2;`

Result Grid

	id	FirstName	LastNAME	Address
▶	2	Nivetha	R	Hydrabad
✱	NULL	NULL	NULL	NULL

20 • `SELECT FirstName, Address FROM STU_DENTS WHERE address IN ("Hydrabad");`

Result Grid

	FirstName	Address
▶	Nivetha	Hydrabad

GROUP BY, HAVING

SQL File 1* SQL File 3* SQL File 4* SQL File 5* x

1 • `SELECT COUNT(id), OriginalName`
 2 `FROM FRIENDS`
 3 `GROUP BY age`
 4 `HAVING COUNT(id) > 0`
 5 `ORDER BY OriginalName DESC;`

Result Grid

	COUNT(id)	OriginalName
▶	1	Mathew Perry
	1	Lisa Kudrow
	1	Jennifer Aniston
	1	David Schwimmer
	2	Courteney Cox

SUM and AVERAGE

SQL File 1* SQL File 3* SQL File 4* SQL File 5* SQL File 6* SQL File 7* x

Limit to 1000 rows

```
1 • SELECT SUM(age)
2 FROM FRIENDS;
```

Result Grid

SUM(age)
323

Filter Rows: | Export: | Wrap Cell Content: |

Result Grid
Form Editor

SQL File 1* SQL File 3* SQL File 4* SQL File 5* SQL File 6* SQL File 7* x

Limit to 1000 rows

```
1 • SELECT AVG(age)
2 FROM FRIENDS;
```

Result Grid

AVG(age)
53.833333333333336

Filter Rows: | Export: | Wrap Cell Content: |

COUNT

SQL File 1* SQL File 3* SQL File 4* SQL File 5* x

Limit to 1000 rows

```
1 • SELECT COUNT(id), OriginalName
2 FROM FRIENDS
3 GROUP BY age
4 HAVING COUNT(id) > 0
5 ORDER BY OriginalName DESC;
```

Result Grid

COUNT(id)	OriginalName
1	Mathew Perry
1	Lisa Kudrow
1	Jennifer Aniston
1	David Schwimmer
2	Courteney Cox

Filter Rows: | Export: | Wrap Cell Content: |

JOINS

INNER JOIN

The screenshot shows a SQL editor window titled 'SQL File 6' with the following query:

```
1 SELECT STU_DENTS.id, FRIENDS.OriginalName
2 FROM STU_DENTS
3 INNER JOIN FRIENDS ON STU_DENTS.id = FRIENDS.id;
```

Below the query editor, the 'Result Grid' is displayed with the following data:

	id	OriginalName
▶	1	Courtney Cox
	2	Jennifer Aniston
	3	Lisa Kudrow

LEFT JOIN

The screenshot shows a SQL editor window titled 'SQL File 6' with the following query:

```
1 SELECT STU_DENTS.id, FRIENDS.OriginalName
2 FROM STU_DENTS
3 LEFT JOIN FRIENDS ON STU_DENTS.id = FRIENDS.id
4 ORDER BY FRIENDS.OriginalName desc;
```

Below the query editor, the 'Result Grid' is displayed with the following data:

	id	OriginalName
▶	3	Lisa Kudrow
	2	Jennifer Aniston
	1	Courtney Cox

RIGHT JOIN

The screenshot shows a SQL IDE interface with a query editor and a result grid. The query editor contains the following SQL code:

```
1 • SELECT STU_DENTS.id, FRIENDS.OriginalName
2 FROM STU_DENTS
3 RIGHT JOIN FRIENDS ON STU_DENTS.id = FRIENDS.id
4 ORDER BY FRIENDS.id ;
```

Below the query editor, the result grid is displayed. It has a toolbar with options like 'Filter Rows', 'Export', and 'Wrap Cell Content'. The result grid shows the following data:

	id	OriginalName
▶	1	Courteney Cox
	2	Jennifer Aniston
	3	Lisa Kudrow
	NULL	Mathew Perry
	NULL	Matt Leblanc
	NULL	David Schwimmer