ON DELETE SET NULL

select * from branch;

branch_id	branch_name	mgr_id	mgr_start_date
1	Corporate	100 [->]	2006-02-09
2	Scranton	*102*	1992-04-06
3	Stamford	106 [->]	1998-02-13
4	Buffalo	*NULL*	*NULL*

delete from employee

where emp_id=102;

select * from branch;

branch_id	branch_name	mgr_id	mgr_start_date
1	Corporate	100 [->]	2006-02-09
2	Scranton	*NULL*	1992-04-06
3	Stamford	106 [->]	1998-02-13
4	Buffalo	*NULL*	*NULL*

mgr_id is the foreign key of employee table hence when we delete then it will update null because when we declare foreign then we set on delete set null.

select * from employee;

emp_id	first_name	last_name	birth_day	sex	salary	super_id	branch_id
100	David	Wallace	1967-11-17	M	250000	*NULL*	1 [->]
101	Jan	Levinson	1961-05-11	F	110000	100 [->]	1 [->]
103	Angela	Martin	1971-06-25	F	63000	*NULL*	2 [->]
104	Kelly	Kapoor	1980-02-05	F	55000	*NULL*	2 [->]
105	Stanley	Hudson	1958-02-19	M	69000	*NULL*	2 [->]
106	Josh	Porter	1969-09-05	M	78000	100 [->]	3 [->]

emp_id	first_name	last_name	birth_day	sex	salary	super_id	branch_id
107	Andy	Bernard	1973-07-22	M	65000	106 [->]	3 [->]
108	Jim	Halpert	1978-10-01	M	71000	106 [->]	3 [->]

ON DELETE SET CASCADE

```
CREATE TABLE branch_supplier (
  branch_id INT,
  supplier_name VARCHAR(40),
  supply_type VARCHAR(40),
  PRIMARY KEY(branch_id, supplier_name),
  FOREIGN KEY(branch_id) REFERENCES branch(branch_id) ON DELETE CASCADE
);
```

```
select * from branch_supplier;
```

branch_id	supplier_name	supply_type
2 [->]	Hammer Mill	Paper
2 [->]	J.T. Forms & Labels	Custom Forms
2 [->]	Uni-ball	Writing Utensils
3 [->]	Hammer Mill	Paper
3 [->]	Patriot Paper	Paper
3 [->]	Stamford Lables	Custom Forms
3 [->]	Uni-ball	Writing Utensils

```
use testdb;
delete from branch
where branch_id=2;
```

```
use testdb;
select * from branch_supplier;
```

branch_id	supplier_name	supply_type
3 [->]	Hammer Mill	Paper
3 [->]	Patriot Paper	Paper
3 [->]	Stamford Lables	Custom Forms
3 [->]	Uni-ball	Writing Utensils

Trigger

Trigger is the block of code which perform some action when we perform some operation in the database.

```
-- CREATE
--     TRIGGER `event_name` BEFORE/AFTER INSERT/UPDATE/DELETE
--     ON `database`.`table`
--     FOR EACH ROW BEGIN
--                 -- trigger body
--                 -- this code is applied to every
--                 -- inserted/updated/deleted row
--     END;

CREATE TABLE trigger_test (
    message VARCHAR(100)
);


DELIMITER $$
CREATE
    TRIGGER my_trigger BEFORE INSERT
    ON employee
    FOR EACH ROW BEGIN
        INSERT INTO trigger_test VALUES('added new employee');
    END$$
DELIMITER ;
INSERT INTO employee
VALUES(109, 'Oscar', 'Martinez', '1968-02-19', 'M', 69000, 106, 3);
```

SELECT * FROM trigger_test;

message
added new employee

DROP Trigger

```
Drop trigger my_trigger;


DELIMITER $$
CREATE
    TRIGGER my_trigger BEFORE INSERT
    ON employee
    FOR EACH ROW BEGIN
        INSERT INTO trigger_test VALUES(NEW.first_name);
    END$$
DELIMITER ;
INSERT INTO employee
VALUES(110, 'Kevin', 'Malone', '1978-02-19', 'M', 69000, 106, 3);
```

SELECT * FROM trigger_test;

message
added new employee
Kevin

```
Drop trigger my_trigger;


DELIMITER $$
CREATE
    TRIGGER my_trigger BEFORE INSERT
    ON employee
```

```
    FOR EACH ROW BEGIN
        IF NEW.sex = 'M' THEN
            INSERT INTO trigger_test VALUES('added male employee');
        ELSEIF NEW.sex = 'F' THEN
            INSERT INTO trigger_test VALUES('added female');
        ELSE
            INSERT INTO trigger_test VALUES('added other employee');
        END IF;
    END$$
DELIMITER ;
INSERT INTO employee
VALUES(111, 'Pam', 'Beesly', '1988-02-19', 'F', 69000, 106, 3);

SELECT * FROM trigger_test;
```

message
added new employee
Kevin
added female

```
DROP TRIGGER my_trigger;
```