mysql

Create Table Syntax

```
CREATE TABLE table_name (
    column1 datatype,
    column2 datatype,
    column3 datatype,
   ....
);
```

String Datatypes in MySQL

1. CHAR(size) 0 to 255
2. VARCHAR(size) 0 to 65535
3. BINARY(size)
4. VARBINARY(size)
5. TINYTEXT 255 characters
6. TEXT(size) 65,535 bytes
7. MEDIUMTEXT 16,777,215 characters
8. LONGTEXT 4,294,967,295 characters
9. TINYBLOB 255 bytes
10. BLOB(size) 65,535 bytes
11. MEDIUMBLOB 16,777,215 bytes
12. LONGBLOB 4,294,967,295 bytes
13. ENUM(val1, val2, val3, ...) list up to 65535 values
14. SET(val1, val2, val3, ...) list up to 64 values

Numeric Datatypes in MySQL

1. BIT(size) 1 to 64
2. TINYINT(size) -128 to 127
3. INT(size) -2147483648 to 2147483647
4. INTEGER(size)
5. SMALLINT(size) -32768 to 32767
6. MEDIUMINT(size) -8388608 to 8388607
7. BIGINT(size) -9223372036854775808 to 9223372036854775807
8. BOOL
9. BOOLEAN 0 / 1
10. FLOAT(p)
11. DOUBLE(size, d) 255.568
12. DECIMAL(size, d) Size = 60 , d = 30
13. DEC(size, d)

Date & Time Datatypes in MySQL

1. DATE '1000-01-01' to '9999-12-31'
2. DATETIME(fsp) YYYY-MM-DD hh:mm:ss
3. TIMESTAMP(fsp)
4. TIME(fsp) hh:mm:ss
5. YEAR four-digit format : 1901

How to Insert data in Tables with SQL ?

Name	Age	Gender
Ram Kumar	21	Male
Salman Khan	22	Male
Meera Khan	21	Female
Sarita Kumari	21	Female
Anil Kapoor	22	Male

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

Insert Multiple Rows Syntax :

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

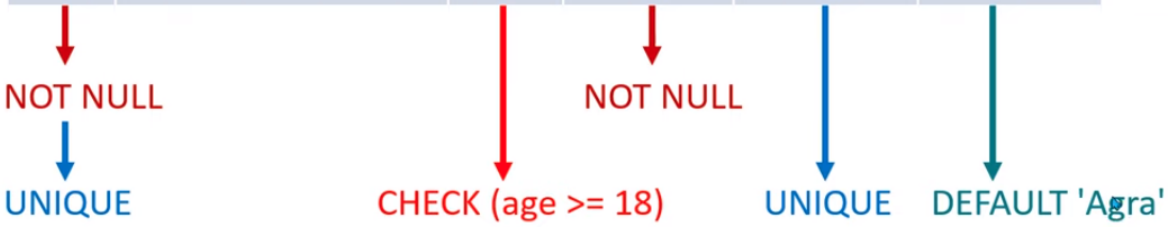
List of Constraints in MySQL

- NOT NULL
- UNIQUE
- DEFAULT
- CHECK
- FOREIGN KEY
- PRIMARY KEY

Data Table without Constraints

Id	Name	Age	Gender	Phone	City
1	Ram Kumar	17	Male	4022155	Agra
2	Salman Khan	19		4033244	Agra
3	Meera Khan	20	Female	4022155	Agra
	Sarita Kumari	18	Female	4066899	Agra
5	Anil Kapoor	19	Male	4188733	Agra

Id	Name	Age	Gender	Phone	City
1	Ram Kumar	17	Male	4022155	Agra
2	Salman Khan	19		4033244	Agra
3	Meera Khan	20	Female	4022155	Agra
	Sarita Kumari	18	Female	4066899	Agra
5	Anil Kapoor	19	Male	4188733	Agra




```
CREATE TABLE table_name (
    id INT NOT NULL UNIQUE,
    name VARCHAR(50) NOT NULL,
    age INT NOT NULL CHECK (age >= 18),
    gender VARCHAR(10) NOT NULL,
    phone VARCHAR(10) NOT NULL UNIQUE,
    city VARCHAR(10) NOT NULL DEFAULT 'Agra',
```

SELECT Syntax

```
SELECT column1, column2, column3, ....
FROM table_name;
```

```
SELECT *
FROM table_name;
```

SELECT with WHERE Clause Syntax

```
SELECT column1, column2, column3, ....
FROM table_name
WHERE condition;
```

WHERE Comparison Operators

Operator	Description
=	Equal
>	Greater than
<	Less than
>=	Greater than or equal
<=	Less than or equal
<> Or !=	Not equal.
BETWEEN	Between a certain range
LIKE	Search for a pattern
IN	To specify multiple possible values for a column

SELECT with AND & OR Operator Syntax

```
SELECT column1, column2, column3, ....
FROM table_name
WHERE condition1 AND condition2 AND condition3 ...;
```

```
SELECT column1, column2, column3, ....
FROM table_name
WHERE condition1 OR condition2 OR condition3 ...;
```

SELECT with IN Operator Syntax

```
SELECT column1, column2, column3, ....
FROM table_name
WHERE column_name IN (value1, value2, ...);

SELECT column1, column2, column3, ....
FROM table_name
WHERE column_name NOT IN (value1, value2, ...);
```

SELECT with BETWEEN Operator Syntax

```
SELECT column1, column2, column3, ....
FROM table_name
WHERE column_name BETWEEN value1 AND value2;

SELECT column1, column2, column3, ....
FROM table_name
WHERE column_name NOT BETWEEN value1 AND value2;
```

LIKE Operator with Wildcard Patterns

Pattern	Description
LIKE 'a%'	Start with "a"
LIKE '%a'	End with "a"
LIKE '%am%'	Have "am" in any position
LIKE 'a%m'	Start with "a" and Ends with "m"
LIKE '_a%'	"a" in the second position
LIKE '__a%'	"a" in the third position
LIKE '_oy'	"o" in the second and "y" in the third position

SELECT with LIKE Operator Syntax

```
SELECT column1, column2, column3, ....
FROM table_name
WHERE column_name LIKE pattern;
```

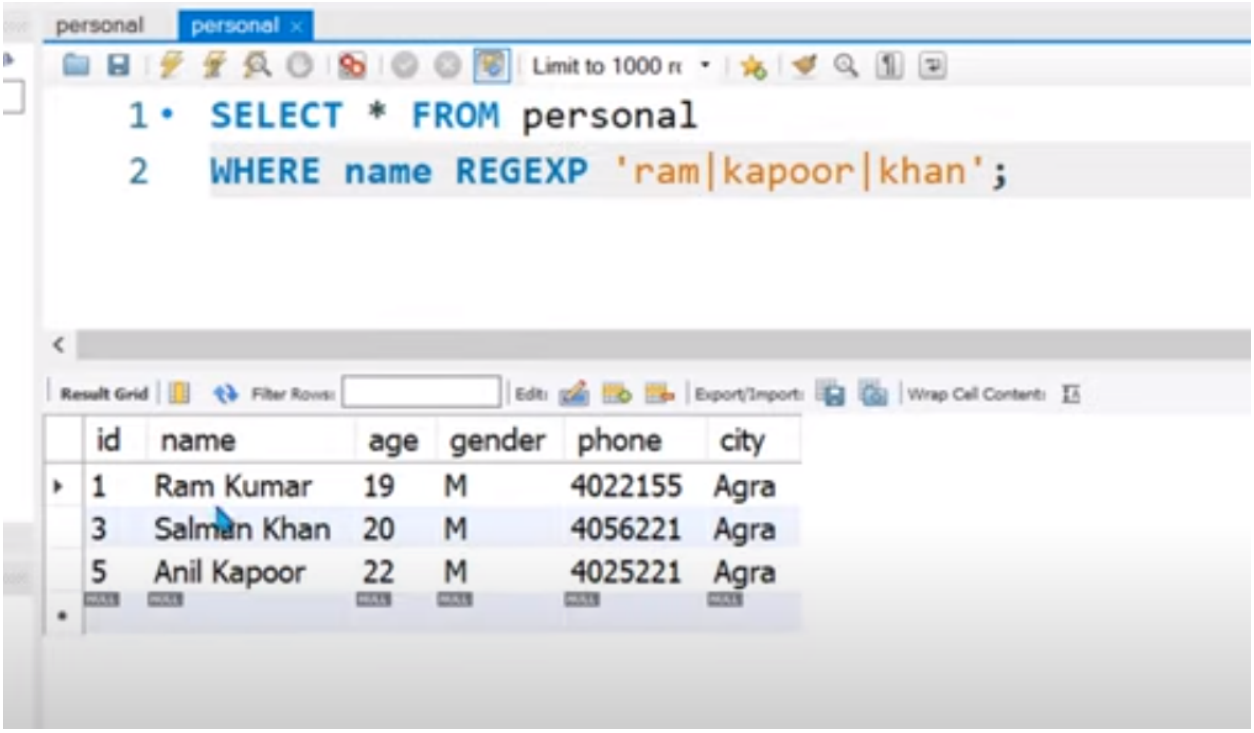
```
SELECT column1, column2, column3, ....
FROM table_name
WHERE column_name NOT LIKE pattern;
```

Regular Expression Patterns with Description

Sign	Pattern	Description
^	'^ra'	Beginning of string
$	'an$'	End of string
[...]	'[rms]'	Any character listed between the square brackets
^[...]	'^[rms]'	Begins with Any character listed between the square brackets
[a-z]	'[a-h]e'	Match with in the range
p1|p2|p3	'tom|dick|harry'	matches any of the patterns p1, p2, or p3

SELECT with Regular Expression Syntax

```
SELECT column1, column2, column3, ....
FROM table_name
WHERE column_name REGEXP pattern;
```

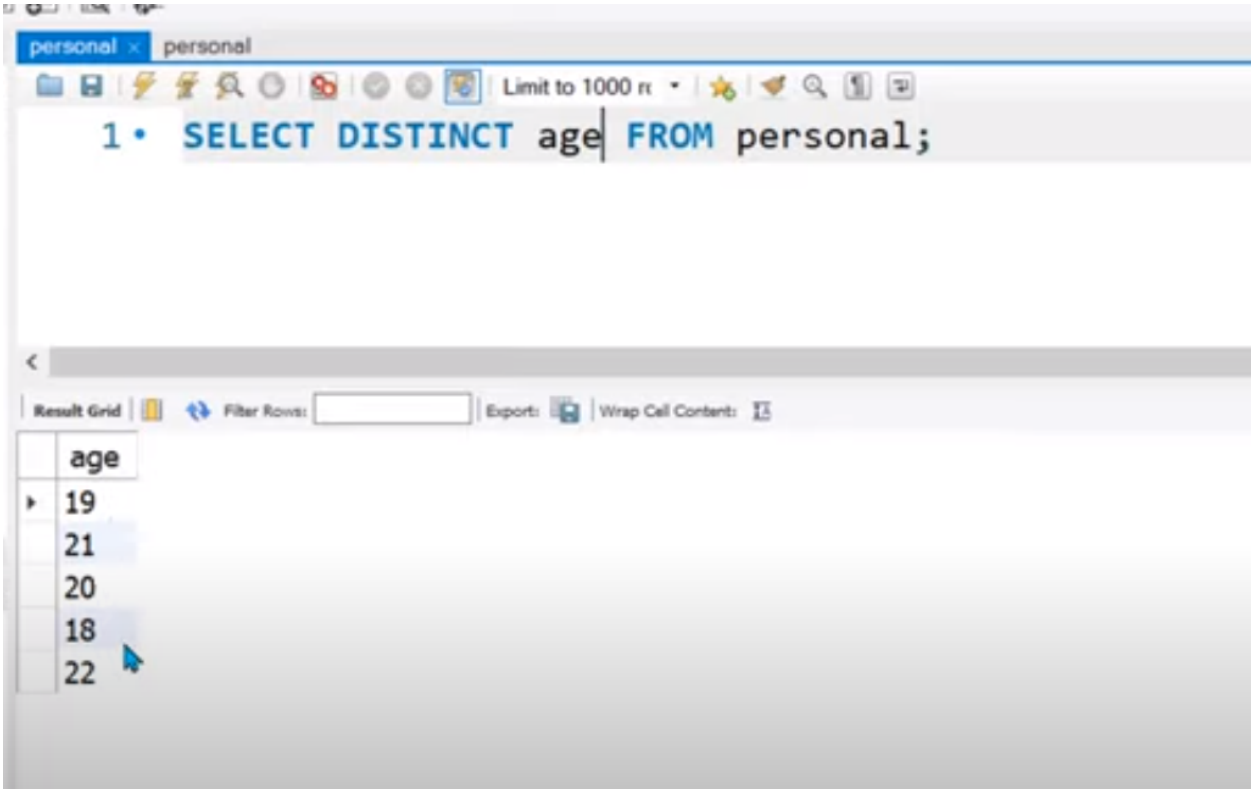


SELECT with ORDER BY Syntax

```
SELECT column1, column2, column3, ....
FROM table_name
ORDER BY column1, column2, .... ASC | DESC;
```

SELECT with DISTINCT Syntax

```
SELECT DISTINCT column1, column2, ....
FROM table_name;
```



SELECT with IS NULL Syntax

```
SELECT column1, column2, column3, ....
FROM table_name
WHERE column IS NULL;
```

```
SELECT column1, column2, column3, ....
FROM table_name
WHERE column IS NOT NULL;
```

SELECT with LIMIT Syntax

```
SELECT column1, column2, column3, ....
FROM table_name
WHERE condition
LIMIT number;
```

SELECT Data with LIMIT & OFFSET

Student Table

Name	Age	Gender
Ram Kumar	19	Male
Salman Khan	18	Male
Meera Khan	20	Female
Sarita Kumari	21	Female
Anil Kapoor	20	Male
Shahid Kapoor	19	Male
Virat Kohli	21	Male

LIMIT 3

Name	Age	Gender
Ram Kumar	19	Male
Salman Khan	18	Male
Meera Khan	20	Female

LIMIT 3, 3

OFFSET LIMIT NUMBER

SELECT with LIMIT & OFFSET Syntax

```
SELECT column1, column2, column3, ....
FROM table_name
WHERE condition
LIMIT offset , number;
```

SELECT Data with Aggregate Functions

Employee Table

Name	Age	Gender	Salary
Ram Kumar	19	Male	4500
Salman Khan	18	Male	5200
Meera Khan	20	Female	6000
Sarita Kumari	21	Female	8500
Anil Kapoor	20	Male	6300
Shahid Kapoor	19	Male	4800
Virat Kohli	21	Male	5700

COUNT(column_name)

MAX(column_name)

MIN(column_name)

SUM(column_name)

AVG(column_name)

SELECT with Aggregate Functions Syntax

```
SELECT COUNT(column_name)
FROM table_name
WHERE condition;

SELECT SUM(column_name)
FROM table_name
WHERE condition;
```

UPDATE Syntax

```
UPDATE table_name
SET column1_name = value1, column2_name = value2,...
WHERE condition;
```

How to Rollback your work in MySQL ?

Employee Table

Id	Name	Age	Salary
1	Ram Kumar	19	4500
2	Salman Khan	18	5200
3	Sarita Kumari	21	8500
4	Anil Kapoor	20	6300

← 6000

```
UPDATE employee
SET Salary = 6000
WHERE Id = 2;

ROLLBACK;
```

How to Rollback your work in MySQL ?

Employee Table

Id	Name	Age	Salary
1	Ram Kumar	19	4500
2	Salman Khan	18	5200
3	Sarita Kumari	21	8500
4	Anil Kapoor	20	6300

6000

22

```
UPDATE employee
SET Age = 22
WHERE Id = 3;

COMMIT;

UPDATE employee
SET Salary = 6000
WHERE Id = 2;

ROLLBACK;
```

COMMIT & ROLLBACK Works for :

- INSERT
- UPDATE
- DELETE

DELETE Syntax

```
DELETE FROM table_name
WHERE condition;
```

```
DELETE FROM table_name;
```

What is PRIMARY KEY Constraint ?

- Primary key always has unique data.
- A primary key cannot have null value.
- A table can contain only one primary key constraint.

Create Table with PRIMARY KEY Syntax

```
CREATE TABLE table_name (
    id INT NOT NULL AUTO_INCREMENT,
    name VARCHAR(50) NOT NULL,
    age INT NOT NULL,
    city VARCHAR(10) NOT NULL ,
    PRIMARY KEY (id)
);
```

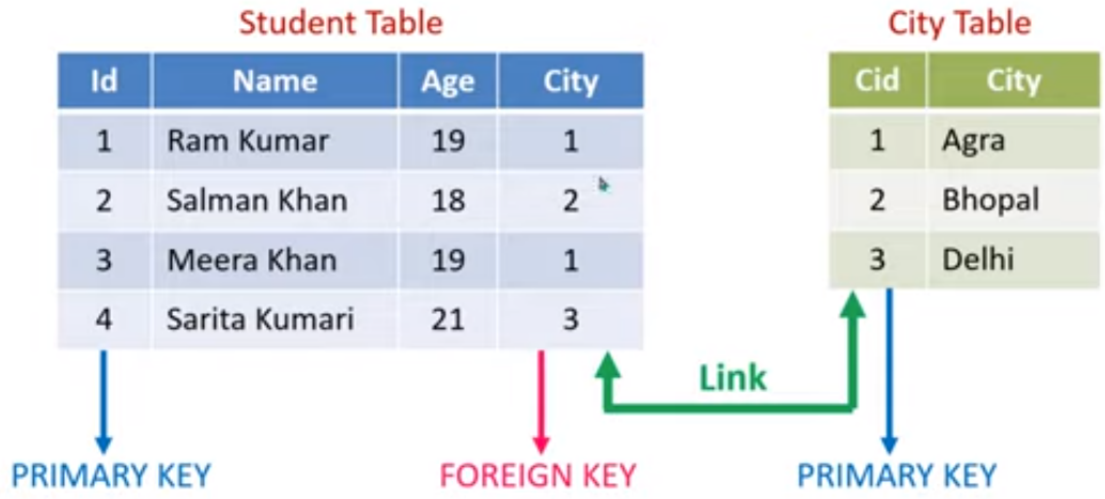
Alter Table with PRIMARY KEY Syntax

```
ALTER TABLE table_name
ADD PRIMARY KEY (id);
```

What is FOREIGN KEY Constraint ?

- A FOREIGN KEY is a key used to link two tables together.
- A FOREIGN key in one table used to point PRIMARY key in another table.

FOREIGN KEY in Table



Create Table with FOREIGN KEY Syntax

```
CREATE TABLE student(
   id INT NOT NULL AUTO_INCREMENT,
   name VARCHAR(50) NOT NULL,
   age INT NOT NULL,
   city VARCHAR(10) NOT NULL ,
   PRIMARY KEY (id),
   FOREIGN KEY (city) REFERENCES City (cid)
);
```

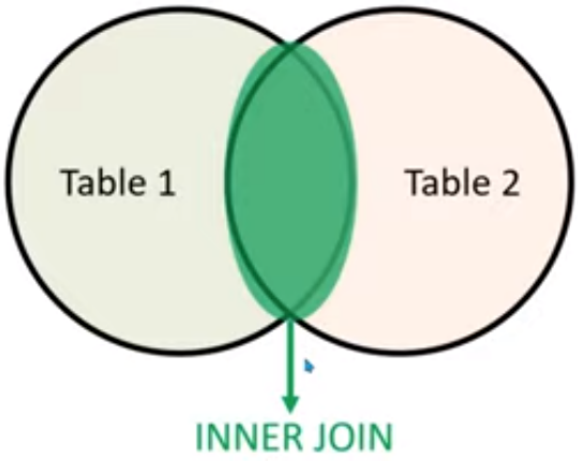
Alter Table with FOREIGN KEY Syntax

```
ALTER TABLE table_name
ADD FOREIGN KEY (city) REFERENCES City (cid);
```

Types of JOINS in MySQL

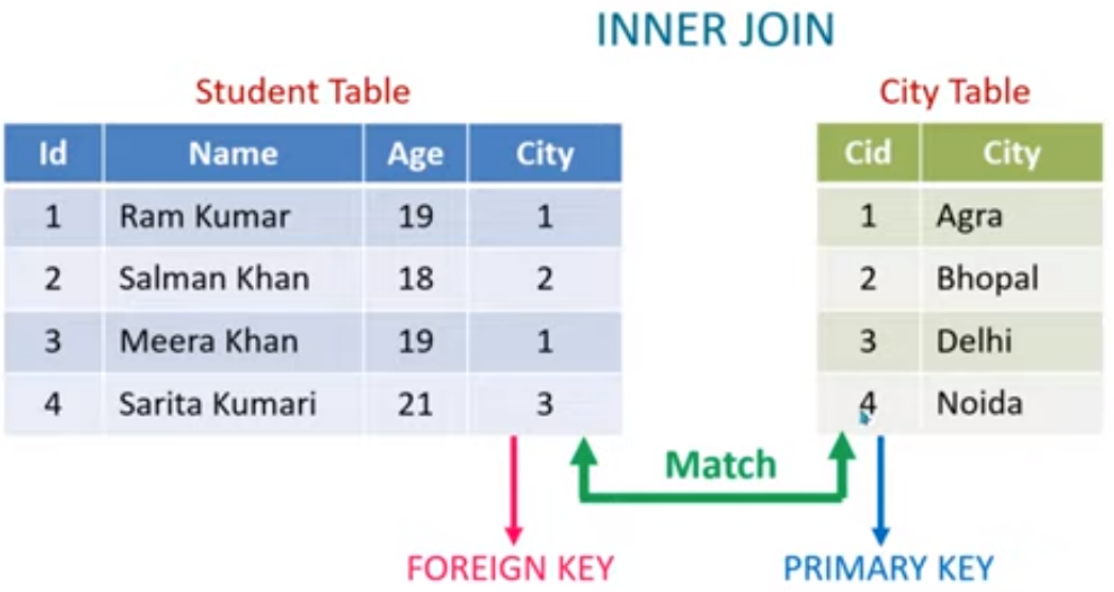
- INNER JOIN
- LEFT JOIN
- RIGHT JOIN
- CROSS JOIN

What is INNER JOIN ?



The INNER JOIN selects records that have matching values in both tables.

JOIN Two Tables



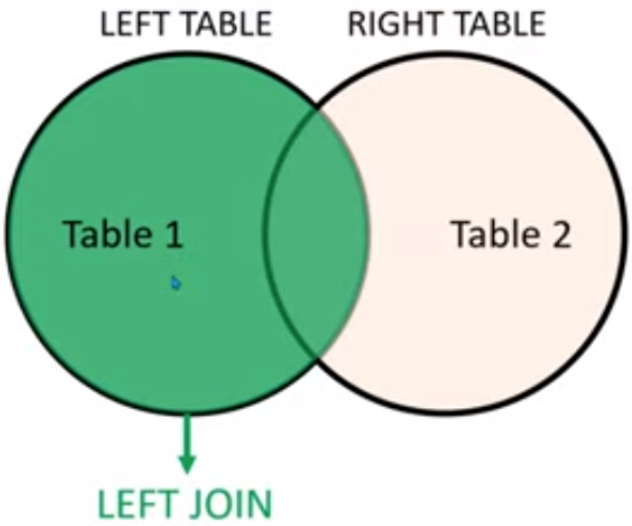
Id	Name	Age	City
1	Ram Kumar	19	1
2	Salman Khan	18	2
3	Meera Khan	19	1
4	Sarita Kumari	21	3

Cid	City
1	Agra
2	Bhopal
3	Delhi
4	Noida

INNER JOIN Syntax

```
SELECT columns
FROM table1
INNER JOIN table2
ON table1.column_name = table2.column_name;
```

What is LEFT JOIN ?



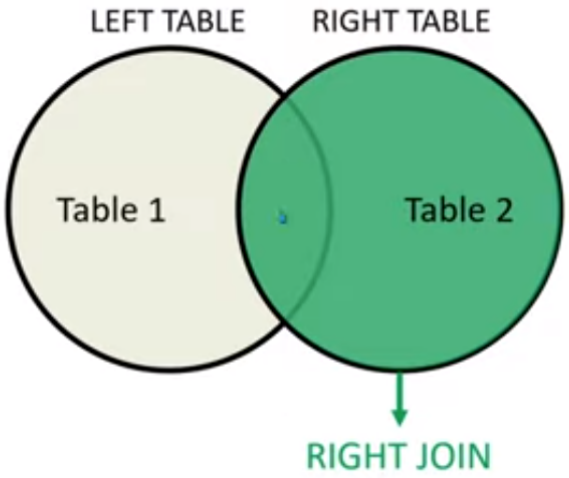
The LEFT JOIN returns all records from the left table (table1),

and the matched records from the right table (table2)

LEFT JOIN Syntax

```
SELECT columns
FROM table1
LEFT JOIN table2
ON table1.column_name = table2.column_name;
```

What is RIGHT JOIN ?



The RIGHT JOIN returns all records from the right table (table2), and the matched records from the left table (table1).

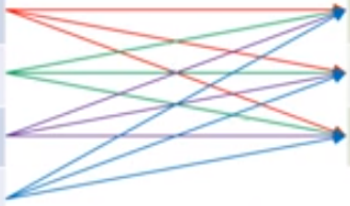
RIGHT JOIN Syntax

```
SELECT columns
FROM table1
RIGHT JOIN table2
ON table1.column_name = table2.column_name;
```

CROSS JOIN Two Tables

Student Table

Id	Name	Age
1	Ram Kumar	19
2	Salman Khan	18
3	Meera Khan	19
4	Sarita Kumari	21



City Table

Cid	City
1	Agra
2	Bhopal
3	Delhi

CROSS JOIN Result

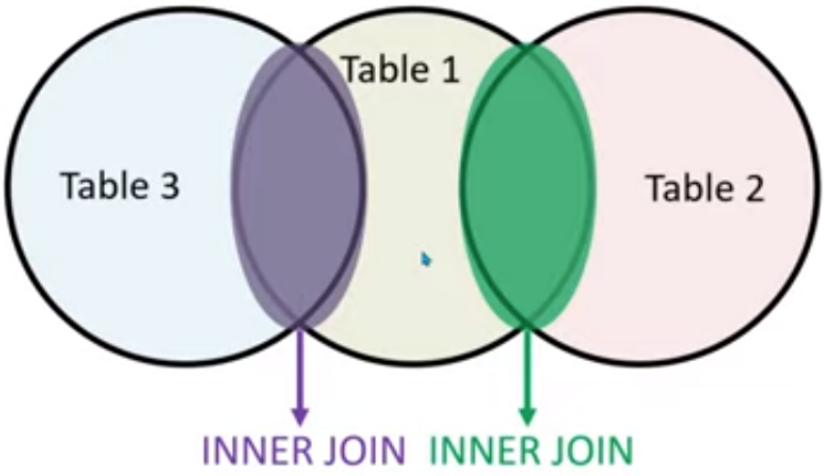
Id	Name	Age	Cid	City
1	Ram Kumar	19	1	Agra
2	Ram Kumar	19	2	Bhopal
3	Ram Kumar	19	3	Delhi
4	Salman Khan	18	1	Agra
5	Salman Khan	18	2	Bhopal
6	Salman Khan	18	3	Delhi
7	Meera Khan	19	1	Agra
8	Meera Khan	19	2	Bhopal
9	Meera Khan	19	3	Delhi
10	Sarita Kumari	21	1	Agra
11	Sarita Kumari	21	2	Bhopal
12	Sarita Kumari	21	3	Delhi

CROSS JOIN Syntax

```
SELECT columns
FROM table1
CROSS JOIN table2;
```

not used much in real scenario 👆

How to JOIN Multiple Tables ?



JOIN Three Tables

Student Table

Id	Name	Age	Courses	City
1	Ram Kumar	19	1	1
2	Salman Khan	18	3	2
3	Meera Khan	19	1	1
4	Sarita Kumari	21	2	3

City Table

Cid	City
1	Agra
2	Bhopal
3	Delhi
4	Noida

Courses Table

Crid	Course
1	Btech
2	BCA
3	BBA

INNER JOIN Syntax for Multiple Tables

```
SELECT columns
FROM table1
INNER JOIN table2
ON table1.column_name = table2.column_name
INNER JOIN table3
ON table1.column_name = table3.column_name;
```

PRIMARY KEY

SELECT with GROUP BY Syntax

```
SELECT columns
FROM table_name
WHERE condition
GROUP BY column_name(s);
```