TABLE JOINS

•••

LEFT, RIGHT, and INNER JOINS and How to Use Them

What is a JOIN in SQL

A JOIN clause is used to combine rows from two or more tables, based on one or more related columns between them.

What is a JOIN in SQL

A JOIN clause is used to combine rows from two or more tables, based on one or more related columns between them.

In order for two tables to be JOINed, we must have one or more columns that appropriately link the two tables of data.

Patient

ID	Name	Species	OwnerID
1	Lola	Cat	204
2	Ada	Dog	1
3	Daisy	Dog	33
4	Champ	Ape	98

Appointment

ID	PatientID	VetID	AppointmentTime
1	1	3	2018-08-02 14:00:00
2	4	1	2018-08-03 10:00:00
3	-1	1	2018-08-03 22:00:00
4	2	1	2018-08-07 12:00:00
5	1	3	2018-08-12 15:00:00

Patient

ID	Name	Species	OwnerID
1	Lola	Cat	204
2	Ada	Dog	1
3	Daisy	Dog	33
4	Champ	Ape	98

Appointment

ID	PatientID	VetID	AppointmentTime
1	1	3	2018-08-02 14:00:00
2	4	1	2018-08-03 10:00:00
3	-1	1	2018-08-03 22:00:00
4	2	1	2018-08-07 12:00:00
5	1	3	2018-08-12 15:00:00

Patient

ID	Name	Species	OwnerID
1	Lola	Cat	204
2	Ada	Dog	1
3	Daisy	Dog	33
4	Champ	Ape	98

Appointment

ID	PatientID	VetID	AppointmentTime
1	1	3	2018-08-02 14:00:00
2	4	1	2018-08-03 10:00:00
3	-1	1	2018-08-03 22:00:00
4	2	1	2018-08-07 12:00:00
5	1	3	2018-08-12 15:00:00

Given any distinct record in either of the two tables, is there a record/row in the other table that is available to JOIN to it?

ID	Name	Species	OwnerID	ID	PatientID	VetID	AppointmentTime
1	Lola	Cat	204	1	1	3	2018-08-02 14:00:00
1	Lola	Cat	204	5	1	3	2018-08-12 15:00:00
2	Ada	Dog	1	4	2	1	2018-08-07 12:00:00
3	Daisy	Dog	33	*No match in the Appointment table for PatientID 3*			
4	Champ	Ape	98	2	4	1	2018-08-03 10:00:00
No match in the Patient table for PatientID -1				3	-1	1	2018-08-03 22:00:00

ID	Name	Species	OwnerID
1	Lola	Cat	204
1	Lola	Cat	204
2	Ada	Dog	1
3	Daisy	Dog	33
4	Champ	Ape	98

No match in the Patient table for PatientID -1

ID	PatientID	VetID	AppointmentTime
1	1	3	2018-08-02 14:00:00
5	1	3	2018-08-12 15:00:00
4	2	1	2018-08-07 12:00:00

No match in the Appointment table for PatientID 3

ID	PatientID	VetID	AppointmentTime
2	4	1	2018-08-03 10:00:00
3	-1	1	2018-08-03 22:00:00

Three Possible Record-Matching Scenarios:

ID	Name	Species	OwnerID	ID	PatientID	VetID	AppointmentTime
1	Lola	Cat	204	1	1	3	2018-08-02 14:00:00
1	Lola	Cat	204	5	1	3	2018-08-12 15:00:00
2	Ada	Dog	1	4	2	1	2018-08-07 12:00:00
3	Daisy	Dog	33	*No match in the Appointment table for PatientID 3*			
4	Champ	Ape	98	2	4	1	2018-08-03 10:00:00
No match in the Patient table for PatientID -1				3	-1	1	2018-08-03 22:00:00

Three Possible Record-Matching Scenarios: Scenario 1

ID	Name	Species	OwnerID	ID	PatientID	VetID	AppointmentTime
1	Lola	Cat	204	1	1	3	2018-08-02 14:00:00
1	Lola	Cat	204	5	1	3	2018-08-12 15:00:00
2	Ada	Dog	1	4	2	1	2018-08-07 12:00:00
3	Daisy	Dog	33	*No match in the Appointment table for PatientID 3*			
4	Champ	Ape	98	2	4	1	2018-08-03 10:00:00
No match in the Patient table for PatientID -1				3	-1	1	2018-08-03 22:00:00

The left table has a record that does not have a match in the right table

Three Possible Record-Matching Scenarios: Scenario 2

ID	Name	Species	OwnerID	ID	PatientID	VetID	AppointmentTime
1	Lola	Cat	204	1	1	3	2018-08-02 14:00:00
1	Lola	Cat	204	5	1	3	2018-08-12 15:00:00
2	Ada	Dog	1	4	2	1	2018-08-07 12:00:00
3	Daisy	Dog	33	*No match in the Appointment table for PatientID 3*			
4	Champ	Ape	98	2	4	1	2018-08-03 10:00:00
No match in the Patient table for PatientID -1				3	-1	1	2018-08-03 22:00:00

The right table has a record that does not have a match in the left table

Three Possible Record-Matching Scenarios: Scenario 3

ID	Name	Species	OwnerID	ID	PatientID	VetID	AppointmentTime
1	Lola	Cat	204	1	1	3	2018-08-02 14:00:00
1	Lola	Cat	204	5	1	3	2018-08-12 15:00:00
2	Ada	Dog	1	4	2	1	2018-08-07 12:00:00
3	Daisy	Dog	33	*No match in the Appointment table for PatientID 3*			
4	Champ	Ape	98	2	4	1	2018-08-03 10:00:00
No match in the Patient table for PatientID -1				3	-1	1	2018-08-03 22:00:00

The left and right tables have a record that matches each other

INNER / LEFT / RIGHT JOINs

(note to other SQL experienced developers: FULL OUTER JOIN is not a join option when coding in MySQL.)

The type of join you choose will influence which records that MySQL can return.

Think about the scenarios and decide:

- Do you want Scenario 1 to be included in the result set?
- Do you want Scenario 2 to be included in the result set?
- Do you want Scenario 3 to be included in the result set?

Example JOIN Syntax in MySQL

```
SELECT *
FROM Patient
    <TypeOfJoin> JOIN  Appointment
    ON Patient.ID = Appointment.PatientID
```

INNER JOIN

Only return data where there is a “match” in both tables (Scenario 3)

```
SELECT *
FROM Patient
    INNER JOIN  Appointment
    ON Patient.ID = Appointment.PatientID;
```

Side note: “INNER” JOINs are the default, so you can actually omit the keyword “INNER”

INNER JOIN

Only return data where there is a “match” in both tables (Scenario 3)

ID	Name	Species	OwnerID	ID	PatientID	VetID	AppointmentTime
1	Lola	Cat	204	1	1	3	2018-08-02 14:00:00
1	Lola	Cat	204	5	1	3	2018-08-12 15:00:00
2	Ada	Dog	1	4	2	1	2018-08-07 12:00:00
4	Champ	Ape	98	2	4	1	2018-08-03 10:00:00

LEFT JOIN

Always return data where there is a record in the LEFT table and include any “matching” records from the RIGHT table in the results set

```
SELECT *

FROM Patient

    LEFT JOIN  Appointment

    ON Patient.ID = Appointment.PatientID;
```

LEFT JOIN

Always return a record where there is a record in the LEFT table and include any “matching” records from the RIGHT table in the results set

ID	Name	Species	OwnerID	ID	PatientID	VetID	AppointmentTime
1	Lola	Cat	204	1	1	3	2018-08-02 14:00:00
1	Lola	Cat	204	5	1	3	2018-08-12 15:00:00
2	Ada	Dog	1	4	2	1	2018-08-07 12:00:00
3	Daisy	Dog	33	*NULL*	*NULL*	*NULL*	*NULL*
4	Champ	Ape	98	2	4	1	2018-08-03 10:00:00

RIGHT JOIN

Always return a record where there is a record in the RIGHT table and include any “matching” records from the LEFT table in the results set

```
SELECT *
FROM Patient
    RIGHT JOIN  Appointment
    ON Patient.ID = Appointment.PatientID
```

RIGHT JOIN

Always return a record where there is a record in the RIGHT table and include any “matching” records from the LEFT table in the results set

ID	Name	Species	OwnerID	ID	PatientID	VetID	AppointmentTime
1	Lola	Cat	204	1	1	3	2018-08-02 14:00:00
1	Lola	Cat	204	5	1	3	2018-08-12 15:00:00
2	Ada	Dog	1	4	2	1	2018-08-07 12:00:00
4	Champ	Ape	98	2	4	1	2018-08-03 10:00:00
NULL	*NULL*	*NULL*	*NULL*	3	-1	1	2018-08-03 22:00:00

A Quick Note on RIGHT JOIN

For our purposes, all RIGHT JOINs can be rewritten as LEFT JOINs by reversing the table order in the JOIN clause

```
T1 RIGHT JOIN T2 ON T1.ID = T2.ID
```

is functionally the same as

```
T2 LEFT JOIN T1 ON T2.ID = T1.ID
```

A Quick Note on RIGHT JOIN

So why are there RIGHT JOINs in MySQL?

- To conform to the SQL ANSI standard
- In rare cases, the user specifying RIGHT JOIN helps MySQL better optimize the query
- If you're feeling lazy and don't want to rewrite a query that JOINs multiple tables already

You can see the debate played out here: https://stackoverflow.com/questions/436345/when-or-why-would-you-use-a-right-outer-join-instead-of-left

Adding Complexity: JOINing Three Tables

Patient

ID	Name	Species	OwnerID
1	Lola	Cat	204
2	Ada	Dog	1
3	Daisy	Dog	33
4	Champ	Ape	98

Appointment

ID	PatientID	VetID	AppointmentTime
1	1	3	2018-08-02 14:00:00
2	4	1	2018-08-03 10:00:00
3	-1	1	2018-08-03 22:00:00
4	2	1	2018-08-07 12:00:00
5	1	3	2018-08-12 15:00:00

Vaccination

ID	PatientID	Type	Date
1	3	Rabies	2016-09-02
2	3	Distemper	2016-09-02
3	4	Flu	2014-04-15
4	1	Rabies	2018-08-03

Adding Complexity: JOINing Three Tables

Patient

ID	Name	Species	OwnerID
1	Lola	Cat	204
2	Ada	Dog	1
3	Daisy	Dog	33
4	Champ	Ape	98

Appointment

ID	PatientID	VetID	AppointmentTime
1	1	3	2018-08-02 14:00:00
2	4	1	2018-08-03 10:00:00
3	-1	1	2018-08-03 22:00:00
4	2	1	2018-08-07 12:00:00
5	1	3	2018-08-12 15:00:00

Vaccination

ID	PatientID	Type	Date
1	3	Rabies	2016-09-02
2	3	Distemper	2016-09-02
3	4	Flu	2014-04-15
4	1	Rabies	2018-08-03

Patient

ID	Name	Species	OwnerID
1	Lola	Cat	204
1	Lola	Cat	204
2	Ada	Dog	1
3	Daisy	Dog	33
3	Daisy	Dog	33
4	Champ	Ape	98
No match in the Patient table for PatientID -1			

Appointment

ID	PatientID	VetID	AppointmentTime
1	1	3	2018-08-02 14:00:00
5	1	3	2018-08-12 15:00:00
4	2	1	2018-08-07 12:00:00
No match in the Appointment table for PatientID 3			
No match in the Appointment table for PatientID 3			
2	4	1	2018-08-03 10:00:00
3	-1	1	2018-08-03 22:00:00

Vaccination

ID	PatientID	Type	Date
4	1	Rabies	2018-08-03
4	1	Rabies	2018-08-03
No match in the Vaccination table for PatientID 2			
1	3	Rabies	2016-09-02
2	3	Distemper	2016-09-02
3	4	Flu	2014-04-15
No match in the Vaccination table for PatientID -1			

INNER JOINs

```
SELECT *

FROM Patient

    INNER JOIN  Appointment ON Patient.ID =
Appointment.PatientID

    INNER JOIN  Vaccination ON Patient.ID = Vaccination.PatientID
```

Patient

ID	Name	Species	OwnerID
1	Lola	Cat	204
1	Lola	Cat	204
2	Ada	Dog	1
3	Daisy	Dog	33
3	Daisy	Dog	33
4	Champ	Ape	98
No match in the Patient table for PatientID -1			

Appointment

ID	PatientID	VetID	AppointmentTime
1	1	3	2018-08-02 14:00:00
5	1	3	2018-08-12 15:00:00
4	2	1	2018-08-07 12:00:00
No match in the Appointment table for PatientID 3			
No match in the Appointment table for PatientID 3			
2	4	1	2018-08-03 10:00:00
3	-1	1	2018-08-03 22:00:00

Vaccination

ID	PatientID	Type	Date
4	1	Rabies	2018-08-03
4	1	Rabies	2018-08-03
No match in the Vaccination table for PatientID 2			
1	3	Rabies	2016-09-02
2	3	Distemper	2016-09-02
3	4	Flu	2014-04-15
No match in the Vaccination table for PatientID -1			

INNER JOINs

```
SELECT *

FROM Patient

    INNER JOIN  Appointment ON Patient.ID = Appointment.PatientID

    INNER JOIN  Vaccination ON Appointment.PatientID = Vaccination.PatientID
```

Patient				Appointment				Vaccination			
ID	Name	Species	OwnerID	ID	PatientID	VetID	AppointmentTime	ID	PatientID	Type	Date
1	Lola	Cat	204	1	1	3	2018-08-02 14:00:00	4	1	Rabies	2018-08-03
1	Lola	Cat	204	5	1	3	2018-08-12 15:00:00	4	1	Rabies	2018-08-03
2	Ada	Dog	1	4	2	1	2018-08-07 12:00:00	*No match in the Vaccination table for PatientID 2*			
3	Daisy	Dog	33	*No match in the Appointment table for PatientID 3*				1	3	Rabies	2016-09-02
3	Daisy	Dog	33	*No match in the Appointment table for PatientID 3*				2	3	Distemper	2016-09-02
4	Champ	Ape	98	2	4	1	2018-08-03 10:00:00	3	4	Flu	2014-04-15
No match in the Patient table for PatientID -1				3	-1	1	2018-08-03 22:00:00	*No match in the Vaccination table for PatientID -1*			

Same results because “A Matches B” AND “B Matches C” has the same effect as “A Matches B” AND “A Matches C”

LEFT JOIN Examples

```
SELECT *

FROM Patient

    LEFT JOIN  Appointment ON Patient.ID = Appointment.PatientID

    LEFT JOIN  Vaccination ON Appointment.PatientID = Vaccination.PatientID
```

Patient

ID	Name	Species	OwnerID
1	Lola	Cat	204
1	Lola	Cat	204
2	Ada	Dog	1
3	Daisy	Dog	33
3	Daisy	Dog	33
4	Champ	Ape	98

Appointment

ID	PatientID	VetID	AppointmentTime
1	1	3	2018-08-02 14:00:00
5	1	3	2018-08-12 15:00:00
4	2	1	2018-08-07 12:00:00
NULL	NULL	NULL	NULL
NULL	NULL	NULL	NULL
2	4	1	2018-08-03 10:00:00
3	-1	1	2018-08-03 22:00:00

No match in the Patient table for PatientID -1

Vaccination

ID	PatientID	Type	Date
4	1	Rabies	2018-08-03
4	1	Rabies	2018-08-03
NULL	NULL	NULL	NULL
1	3	Rabies	2016-09-02
2	3	Distemper	2016-09-02
3	4	Flu	2014-04-15

No match in the Vaccination table for PatientID -1

Patient

ID	Name	Species	OwnerID
1	Lola	Cat	204
1	Lola	Cat	204
2	Ada	Dog	1
3	Daisy	Dog	33
3	Daisy	Dog	33
4	Champ	Ape	98

Appointment

ID	PatientID	VetID	AppointmentTime
1	1	3	2018-08-02 14:00:00
5	1	3	2018-08-12 15:00:00
4	2	1	2018-08-07 12:00:00
NULL	NULL	NULL	NULL
NULL	NULL	NULL	NULL
2	4	1	2018-08-03 10:00:00
No match in the Patient table for PatientID -1			
3	-1	1	2018-08-03 22:00:00

Vaccination

ID	PatientID	Type	Date
4	1	Rabies	2018-08-03
4	1	Rabies	2018-08-03
NULL	NULL	NULL	NULL
NULL	NULL	NULL	NULL
NULL	NULL	NULL	NULL
3	4	Flu	2014-04-15
No match in the Vaccination table for PatientID -1			

LEFT JOIN Examples

```
SELECT *

FROM Patient

    LEFT JOIN  Appointment ON Patient.ID = Appointment.PatientID

    LEFT JOIN  Vaccination ON Patient.ID = Vaccination.PatientID
```

Patient

ID	Name	Species	OwnerID
1	Lola	Cat	204
1	Lola	Cat	204
2	Ada	Dog	1
3	Daisy	Dog	33
3	Daisy	Dog	33
4	Champ	Ape	98
No match in the Patient table for PatientID -1			

Appointment

ID	PatientID	VetID	AppointmentTime
1	1	3	2018-08-02 14:00:00
5	1	3	2018-08-12 15:00:00
4	2	1	2018-08-07 12:00:00
NULL	*NULL*	*NULL*	*NULL*
NULL	*NULL*	*NULL*	*NULL*
2	4	1	2018-08-03 10:00:00
3	-1	1	2018-08-03 22:00:00

Vaccination

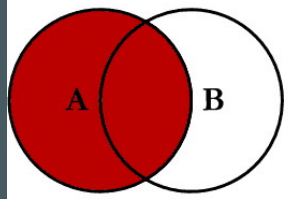
ID	PatientID	Type	Date
4	1	Rabies	2018-08-03
4	1	Rabies	2018-08-03
NULL	*NULL*	*NULL*	*NULL*
1	3	Rabies	2016-09-02
2	3	Distemper	2016-09-02
3	4	Flu	2014-04-15
No match in the Vaccination table for PatientID -1			

Using LEFT and INNER JOINs Together

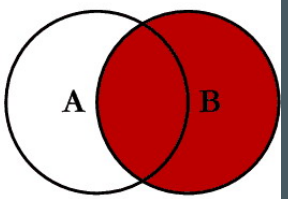
```
SELECT *

FROM Patient

    INNER JOIN  Appointment ON Patient.ID = Appointment.PatientID

    LEFT JOIN  Vaccination ON Appointment.PatientID = Vaccination.PatientID
```

Patient				Appointment				Vaccination			
ID	Name	Species	OwnerID	ID	PatientID	VetID	AppointmentTime	**ID**	**PatientID**	**Type**	**Date**
1	Lola	Cat	204	1	1	3	2018-08-02 14:00:00	4	1	Rabies	2018-08-03
1	Lola	Cat	204	5	1	3	2018-08-12 15:00:00	4	1	Rabies	2018-08-03
2	Ada	Dog	1	4	2	1	2018-08-07 12:00:00	*NULL*	*NULL*	*NULL*	*NULL*
3	Daisy	Dog	33	*No match in the Appointment table for PatientID 3*				1	3	Rabies	2016-09-02
3	Daisy	Dog	33	*No match in the Appointment table for PatientID 3*				2	3	Distemper	2016-09-02
4	Champ	Ape	98	2	4	1	2018-08-03 10:00:00	3	4	Flu	2014-04-15
No match in the Patient table for PatientID -1				3	-1	1	2018-08-03 22:00:00	*No match in the Vaccination table for PatientID -1*			

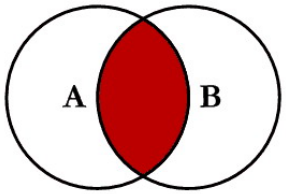
SQL JOINS



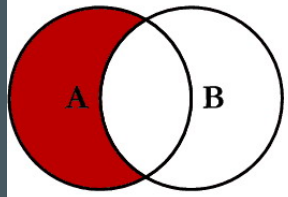

```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
```



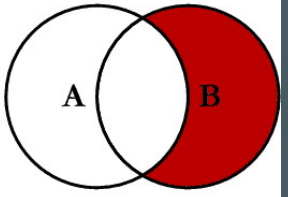

```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
```



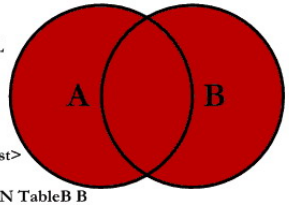

```
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
```



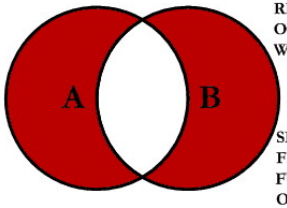

```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL
```




```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
```




```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
```




```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
OR B.Key IS NULL
```

Big Takeaways

- JOINs are incredibly complex; it’s easy to get unintended results, so make sure to test!
- JOINs are powerful!

NULLS and Three-Valued Logic

...

```
SELECT *
FROM Student
```

ID	FirstName	LastName	BirthDate
1	Hope	Smith	1990-11-05
2	Perry	Mason	1993-06-16
3	Hermann	Munster	1989-10-16
4	Lily	Tomlin	NULL

```
SELECT *
FROM Student
WHERE BirthDate = '1990-11-05'
```

ID	FirstName	LastName	BirthDate
1	Hope	Smith	1990-11-05

```
SELECT *
FROM Student
WHERE BirthDate != '1990-11-05'
```

ID	FirstName	LastName	BirthDate
2	Perry	Mason	1993-06-16
3	Hermann	Munster	1989-10-16

How to Test for Null

```
SELECT <columns>
FROM <table_name>
WHERE <columnName> IS NULL
```

```
SELECT <columns>
FROM <table_name>
WHERE <columnName> IS NOT NULL
```

```
SELECT *
FROM Student
WHERE BirthDate IS NULL
```

ID	FirstName	LastName	BirthDate
4	Lily	Tomlin	NULL

```
SELECT *
FROM Student
WHERE BirthDate IS NOT NULL
```

ID	FirstName	LastName	BirthDate
1	Hope	Smith	1990-11-05
2	Perry	Mason	1993-06-16
3	Hermann	Munster	1989-10-16

Case Study - The Danger of Treating NULL Like a Value

```
CREATE TABLE PortfolioValue (
    PortfolioID INT NOT NULL,
    ReportDate DATE NOT NULL,
    CurrencyID INT,
    ConvertedCurrencyID INT,
    PortfolioValue FLOAT NOT NULL,
    ConvertedValue FLOAT NOT NULL
);
```

PortfolioID	ReportDate	CurrencyID	ConvertedCurrencyID	PortfolioValue	ConvertedValue
1	2018-01-01	NULL	NULL	100.01	100.01
1	2018-01-02	NULL	NULL	102.5	102.5
1	2018-01-01	1	2	300.33	425.81
1	2018-01-01	1	2	299.67	420.34
2	2018-01-01	2	4	204.09	180.22
2	2018-01-02	2	4	210.77	190.22
3	2018-01-01	1	1	360.65	360.65
3	2018-01-01	1	1	360.01	360.65

The request is to find all rows where CurrencyID is equal to ConvertedCurrencyID

Please include records where both columns are NULL

PortfolioID	ReportDate	CurrencyID	ConvertedCurrencyID	PortfolioValue	ConvertedValue
1	2018-01-01	NULL	NULL	100.01	100.01
1	2018-01-02	NULL	NULL	102.5	102.5
1	2018-01-01	1	2	300.33	425.81
1	2018-01-01	1	2	299.67	420.34
2	2018-01-01	2	4	204.09	180.22
2	2018-01-02	2	4	210.77	190.22
3	2018-01-01	1	1	360.65	360.65
3	2018-01-01	1	1	360.01	360.65

?

```
SELECT PortfolioID, ReportDate, PortfolioValue
FROM PortfolioValue
WHERE CurrencyID = ConvertedCurrencyID
```

PortfolioID	ReportDate	CurrencyID	ConvertedCurrencyID	PortfolioValue	ConvertedValue
3	2018-01-01	1	1	360.65	360.65
3	2018-01-01	1	1	360.01	360.65

```
SELECT PortfolioID, ReportDate, PortfolioValue
FROM PortfolioValue
WHERE CurrencyID = ConvertedCurrencyID
```

Coping with NULL Option One - Explicitly Check

PortfolioID	ReportDate	CurrencyID	ConvertedCurrencyID	PortfolioValue	ConvertedValue
1	2018-01-01	NULL	NULL	100.01	100.01
1	2018-01-02	NULL	NULL	102.5	102.5
3	2018-01-01	1	1	360.65	360.65
3	2018-01-01	1	1	360.01	360.65

```
SELECT PortfolioID, ReportDate, PortfolioValue
FROM PortfolioValue
WHERE CurrencyID = ConvertedCurrencyID
    OR (CurrencyID IS NULL AND ConvertedCurrencyID IS NULL)
```

Coping with NULL Option Two - Use IFNULL

PortfolioID	ReportDate	CurrencyID	ConvertedCurrencyID	PortfolioValue	ConvertedValue
1	2018-01-01	NULL	NULL	100.01	100.01
1	2018-01-02	NULL	NULL	102.5	102.5
3	2018-01-01	1	1	360.65	360.65
3	2018-01-01	1	1	360.01	360.65

```
SELECT PortfolioID, ReportDate, PortfolioValue
FROM PortfolioValue
WHERE IFNULL(CurrencyID, -999) =
      IFNULL(ConvertedCurrencyID, -999)
```

Coping with NULL Option Three - Rethink Table Design

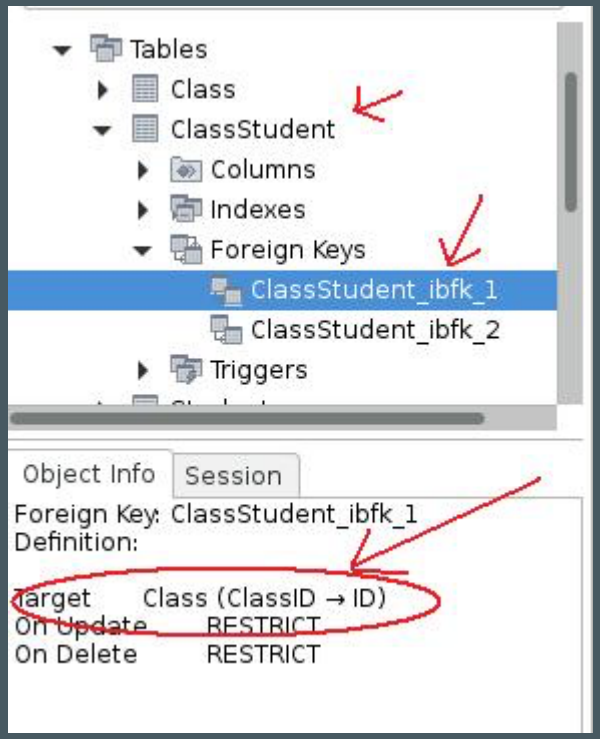
Ask yourself - “Does this column really need to be NULLABLE?”

If not, then add the NOT NULL constraint to the column and save yourself unnecessary headache!!

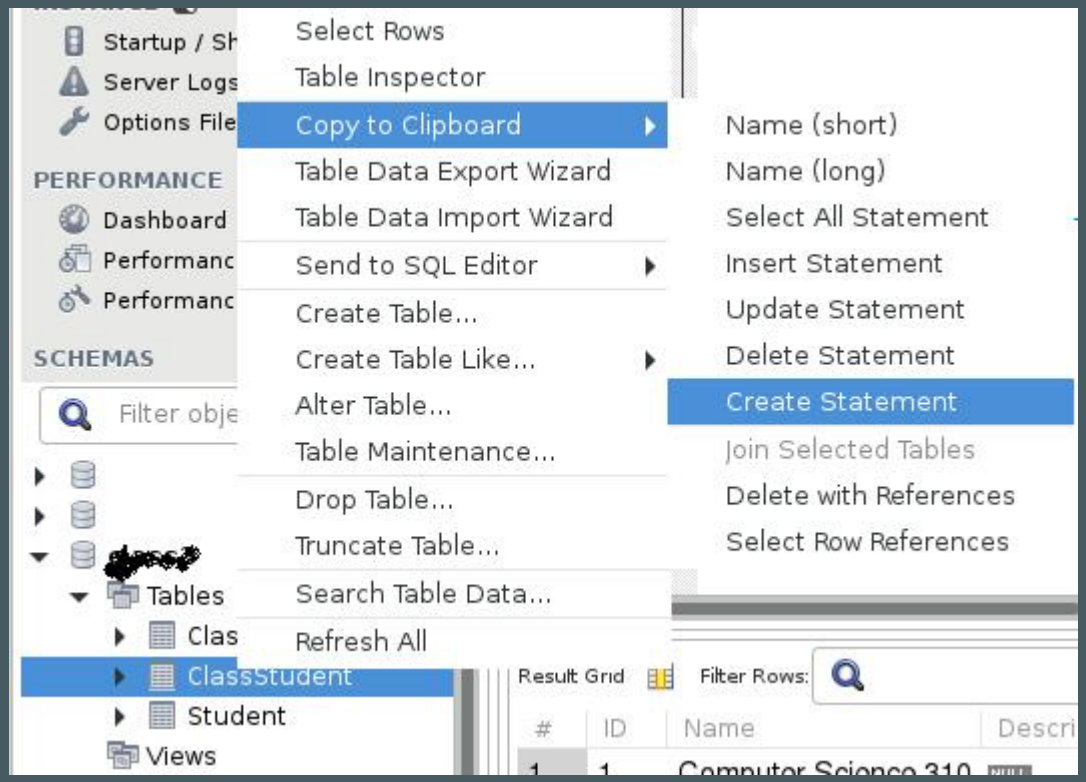
Foreign Key Review

...

Find a Foreign Key - First Way



Find a Foreign Key - Second Way

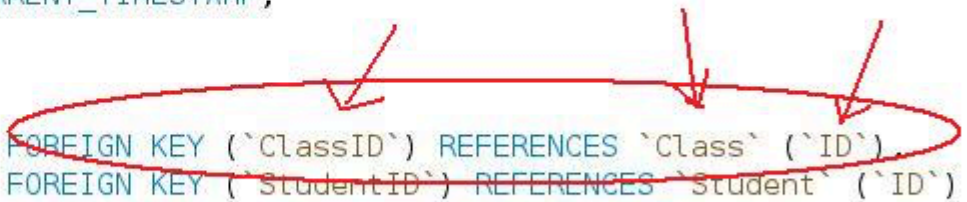


Find a Foreign Key - Second Way

```
1  CREATE TABLE `ClassStudent` (
2    `ID` int(11) NOT NULL AUTO_INCREMENT,
3    `ClassID` int(11) NOT NULL,
4    `StudentID` int(11) NOT NULL,
5    `SignupDate` datetime DEFAULT CURRENT_TIMESTAMP,
6    PRIMARY KEY (`ID`),
7    KEY `ClassID` (`ClassID`),
8    KEY `StudentID` (`StudentID`),
9    CONSTRAINT `ClassStudent_ibfk_1` FOREIGN KEY (`ClassID`) REFERENCES `Class` (`ID`),
10   CONSTRAINT `ClassStudent_ibfk_2` FOREIGN KEY (`StudentID`) REFERENCES `Student` (`ID`)
11 ) ENGINE=InnoDB AUTO_INCREMENT=8 DEFAULT CHARSET=latin1;
12
```

Find a Foreign Key - Second Way

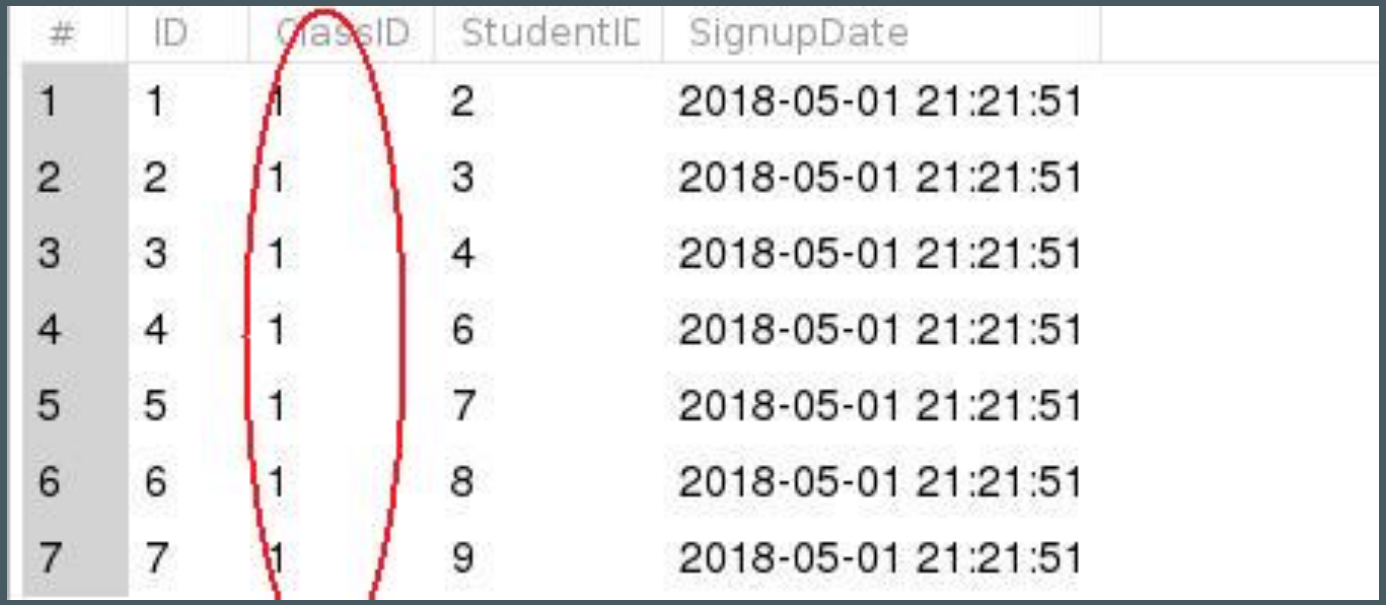
```
1  CREATE TABLE `ClassStudent` (
2    `ID` int(11) NOT NULL AUTO_INCREMENT,
3    `ClassID` int(11) NOT NULL,
4    `StudentID` int(11) NOT NULL,
5    `SignupDate` datetime DEFAULT CURRENT_TIMESTAMP,
6    PRIMARY KEY (`ID`),
7    KEY `ClassID` (`ClassID`),
8    KEY `StudentID` (`StudentID`),
9    CONSTRAINT `ClassStudent_ibfk_1` FOREIGN KEY (`ClassID`) REFERENCES `Class` (`ID`),
10   CONSTRAINT `ClassStudent_ibfk_2` FOREIGN KEY (`StudentID`) REFERENCES `Student` (`ID`)
11 ) ENGINE=InnoDB AUTO_INCREMENT=8 DEFAULT CHARSET=latin1;
12
```



How does it apply? Start by looking at the Table

#	ID	ClassID	StudentID	SignupDate
1	1	1	2	2018-05-01 21:21:51
2	2	1	3	2018-05-01 21:21:51
3	3	1	4	2018-05-01 21:21:51
4	4	1	6	2018-05-01 21:21:51
5	5	1	7	2018-05-01 21:21:51
6	6	1	8	2018-05-01 21:21:51
7	7	1	9	2018-05-01 21:21:51

Now Identify the Column that has the Foreign Key



#	ID	ClassID	StudentID	SignupDate
1	1	1	2	2018-05-01 21:21:51
2	2	1	3	2018-05-01 21:21:51
3	3	1	4	2018-05-01 21:21:51
4	4	1	6	2018-05-01 21:21:51
5	5	1	7	2018-05-01 21:21:51
6	6	1	8	2018-05-01 21:21:51
7	7	1	9	2018-05-01 21:21:51

Now Look at the “references” table

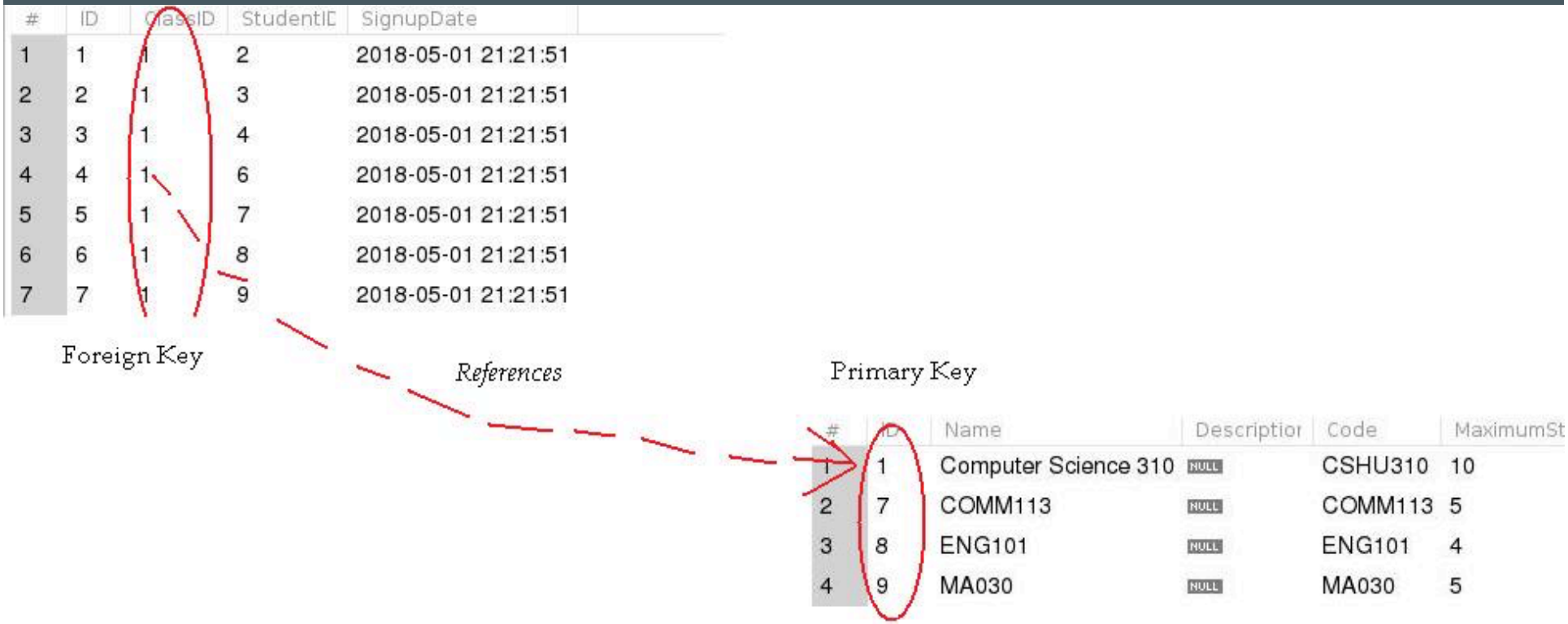
#	ID	Name	Descriptior	Code	MaximumStudents
1	1	Computer Science 310	NULL	CSHU310	10
2	7	COMM113	NULL	COMM113	5
3	8	ENG101	NULL	ENG101	4
4	9	MA030	NULL	MA030	5

Now look at them together

#	ID	ClassID	StudentID	SignupDate
1	1	1	2	2018-05-01 21:21:51
2	2	1	3	2018-05-01 21:21:51
3	3	1	4	2018-05-01 21:21:51
4	4	1	6	2018-05-01 21:21:51
5	5	1	7	2018-05-01 21:21:51
6	6	1	8	2018-05-01 21:21:51
7	7	1	9	2018-05-01 21:21:51

#	ID	Name	Description	Code	MaximumSt
1	1	Computer Science 310	NULL	CSHU310	10
2	7	COMM113	NULL	COMM113	5
3	8	ENG101	NULL	ENG101	4
4	9	MA030	NULL	MA030	5

So…. Which Class are the students taking?



#	ID	ClassID	StudentID	SignupDate
1	1	1	2	2018-05-01 21:21:51
2	2	1	3	2018-05-01 21:21:51
3	3	1	4	2018-05-01 21:21:51
4	4	1	6	2018-05-01 21:21:51
5	5	1	7	2018-05-01 21:21:51
6	6	1	8	2018-05-01 21:21:51
7	7	1	9	2018-05-01 21:21:51

#	ID	Name	Description	Code	MaximumSt
1	1	Computer Science 310	NULL	CSHU310	10
2	7	COMM113	NULL	COMM113	5
3	8	ENG101	NULL	ENG101	4
4	9	MA030	NULL	MA030	5

```
SELECT *
FROM ClassStudent cs                    -- 1. This table's Foreign Key
JOIN  Class c                           -- 2 References this table
        On c.ID = CS.ClassID            -- 3. The "connection" is between these columns
```

```
SELECT *
FROM Class c                                    -- 2 References this table
JOIN ClassStudent cs                            -- 1. This table's Foreign Key
        On c.ID = CS.ClassID                    -- 3. The "connection" is
between these columns
```