

SQL Joins

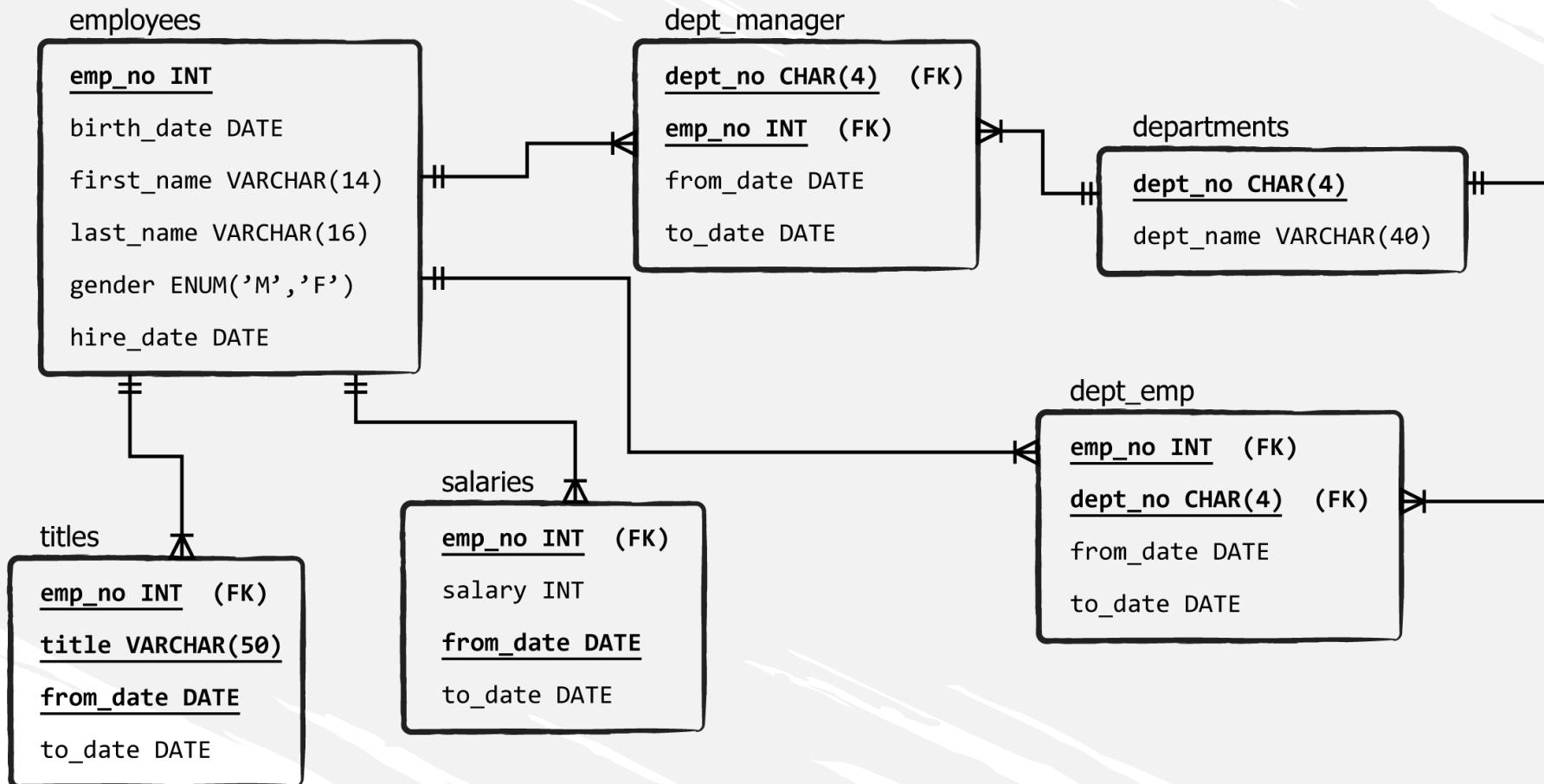
A large conference room with rows of chairs facing a central table. The room has a modern design with large windows overlooking a landscape. The chairs are arranged in several rows, facing towards the front of the room where a long table would be. The room is well-lit by natural light coming through the windows.

What are JOINs?

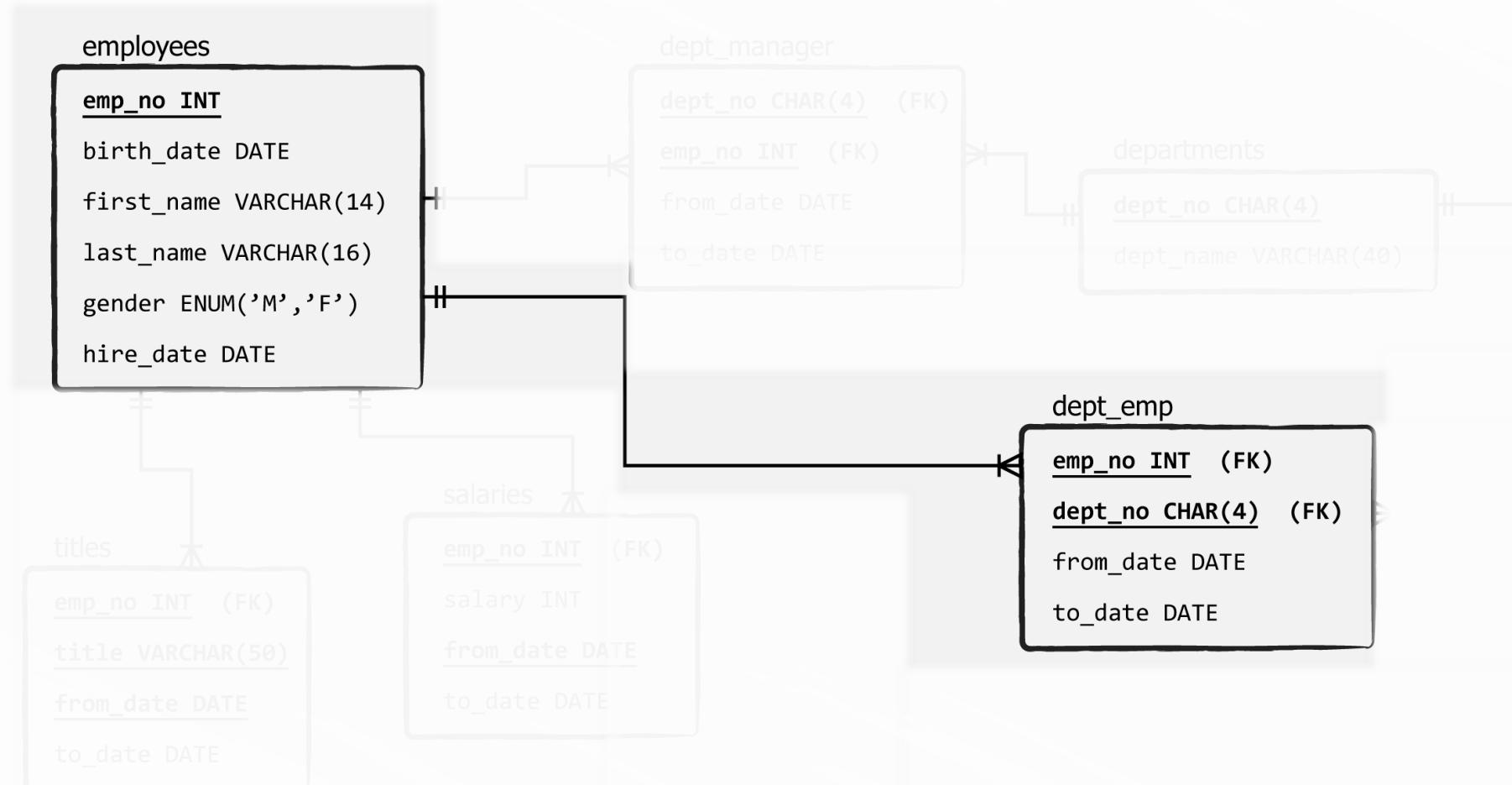
Introduction to Joins

- **joins**
the SQL tool that allow us to construct a relationship between objects

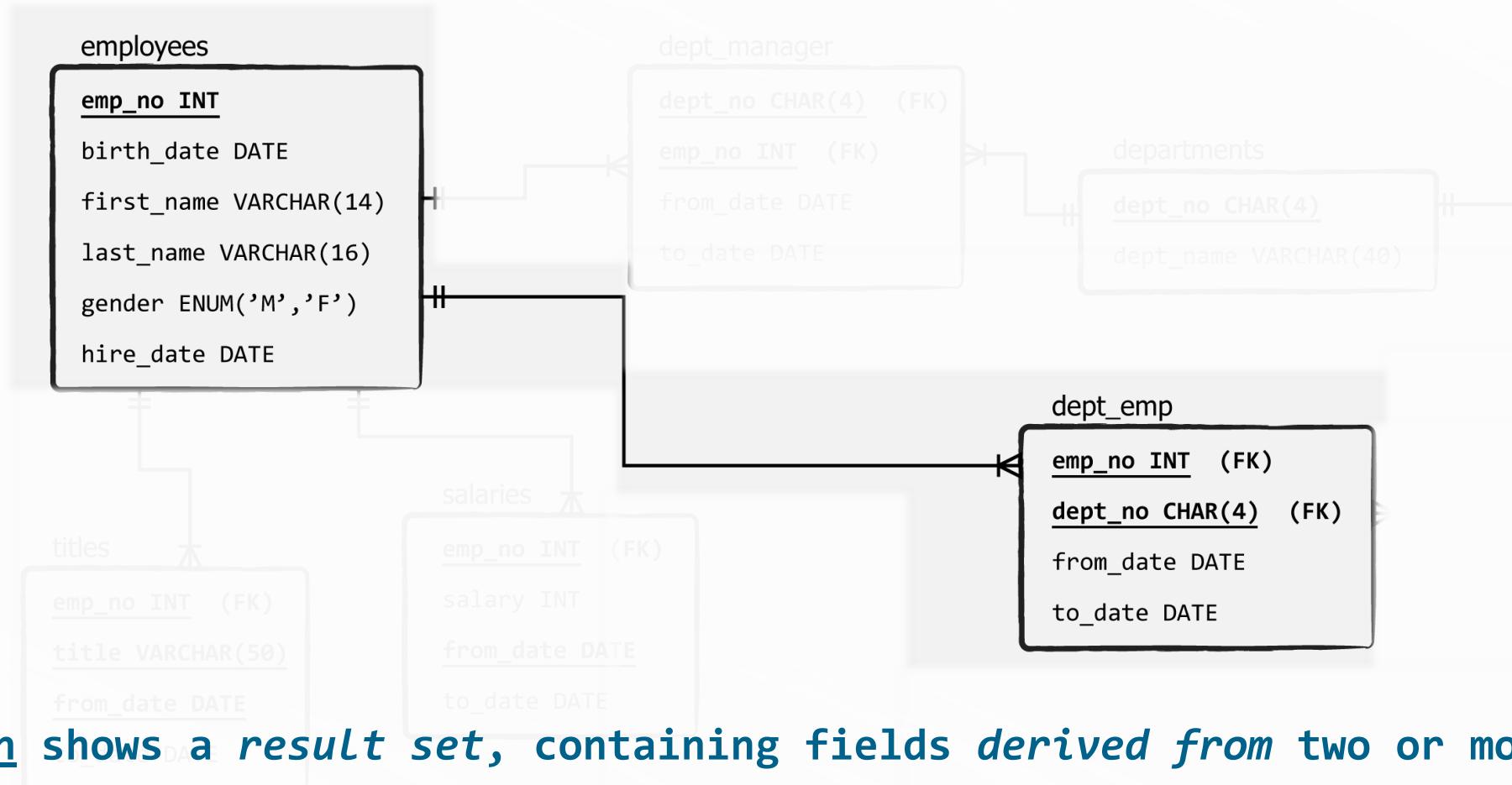
Introduction to Joins



Introduction to Joins



Introduction to Joins



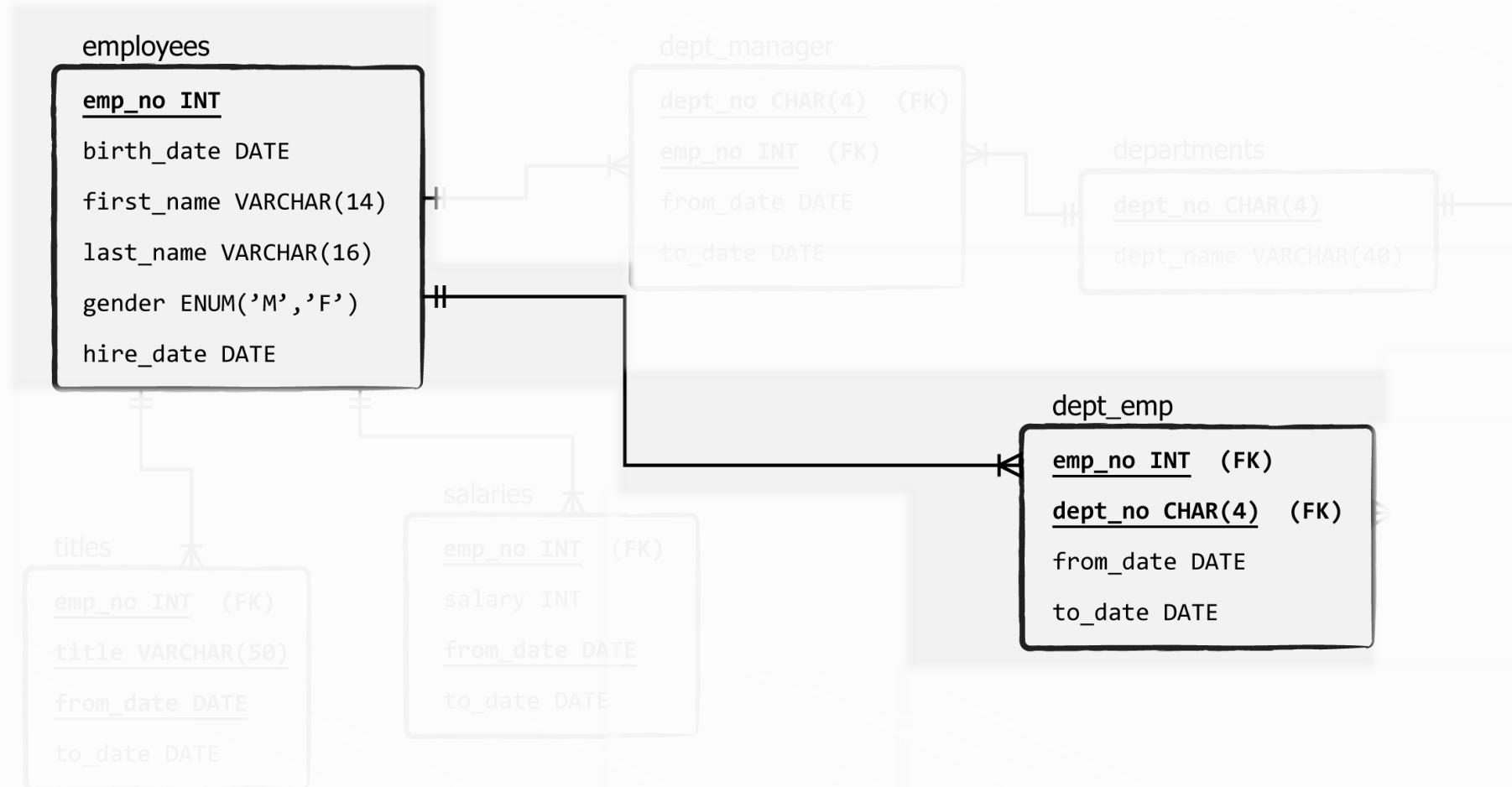
a join shows a *result set*, containing fields *derived from two or more tables*

Introduction to Joins

- **joins**

- we must find a *related column* from the two tables that contains the same *type* of data

Introduction to Joins

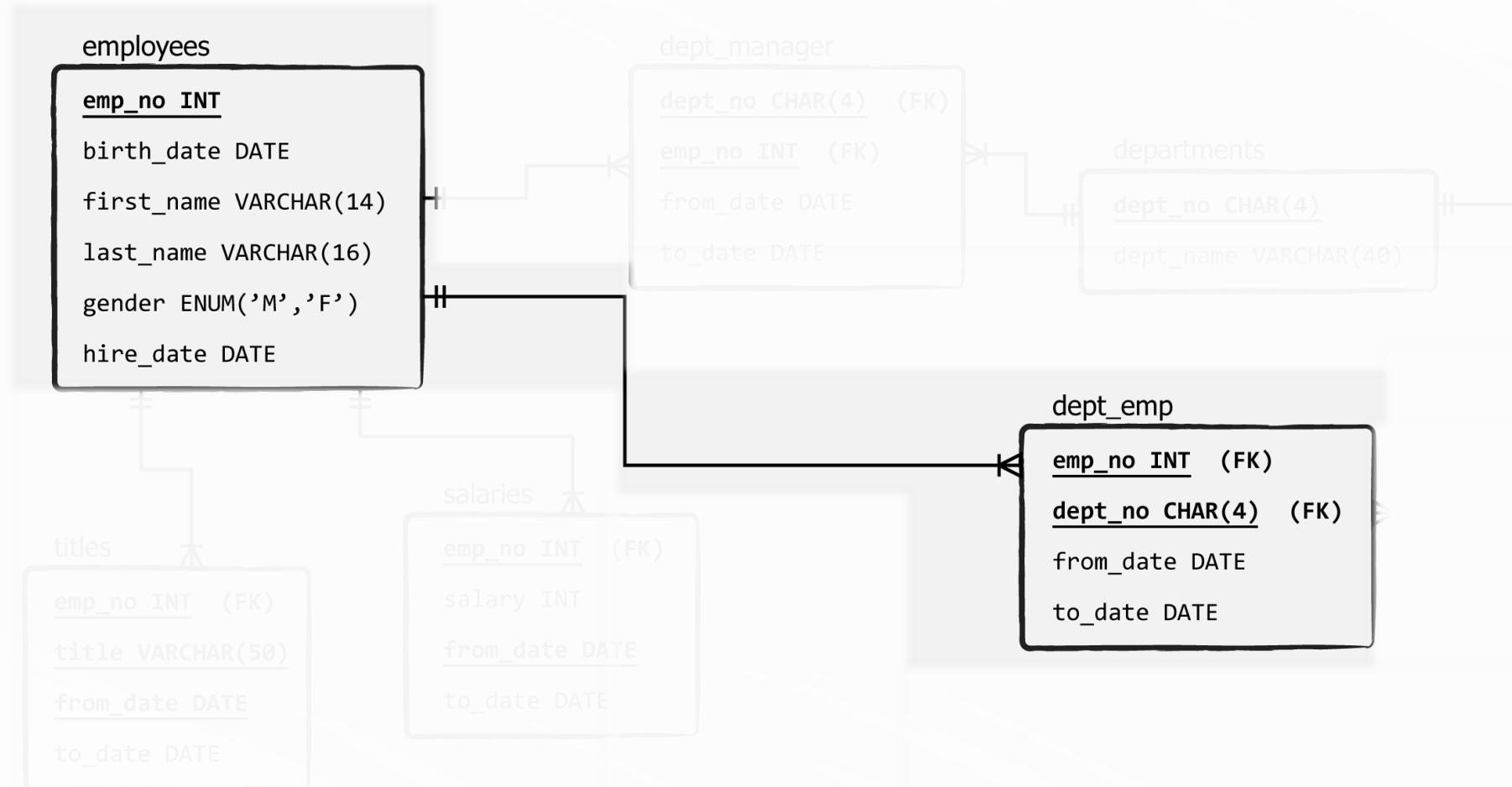


Introduction to Joins

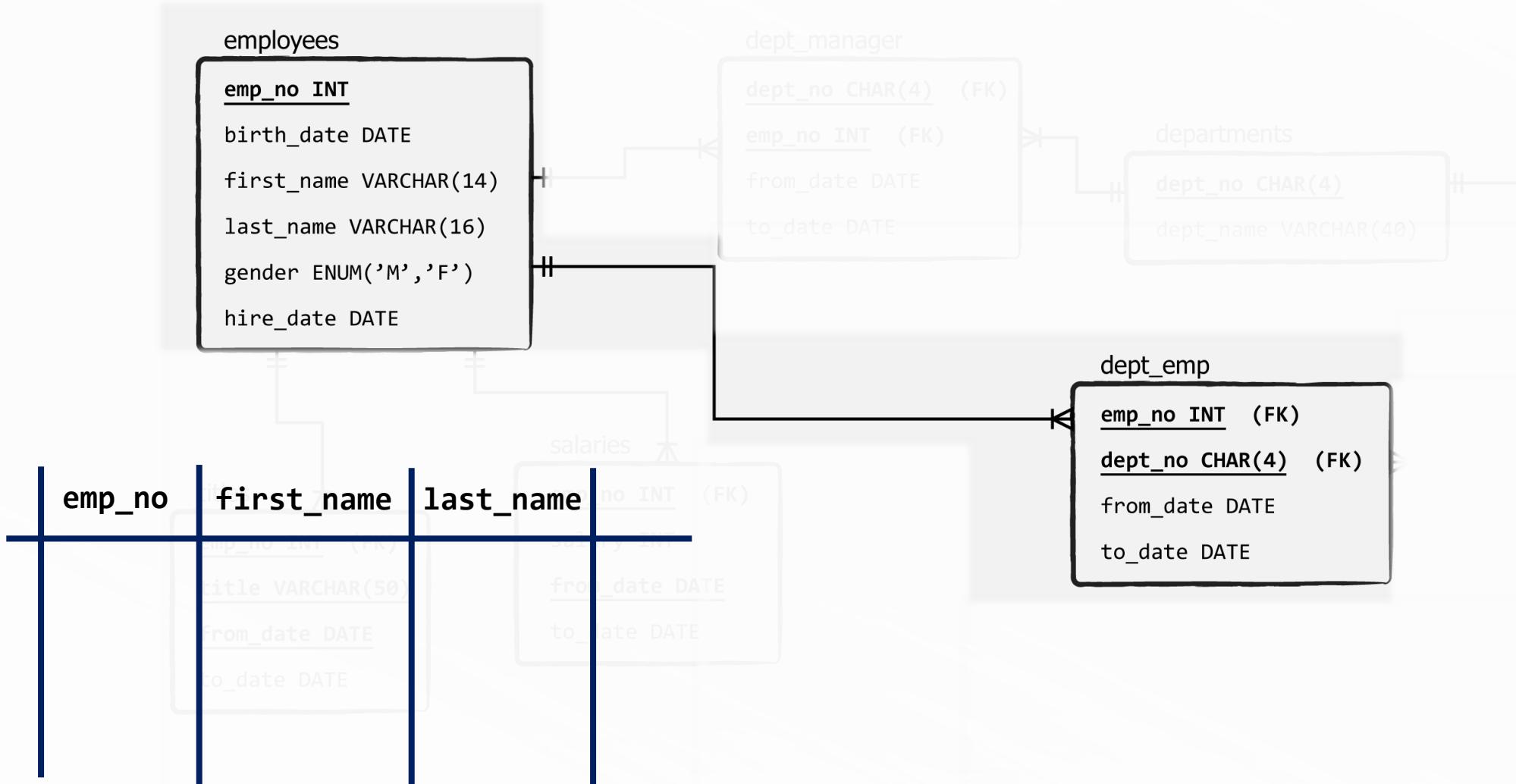
- **joins**

- we must find a *related column* from the two tables that contains the same *type* of data
- we will be free to add columns from these two tables to our output

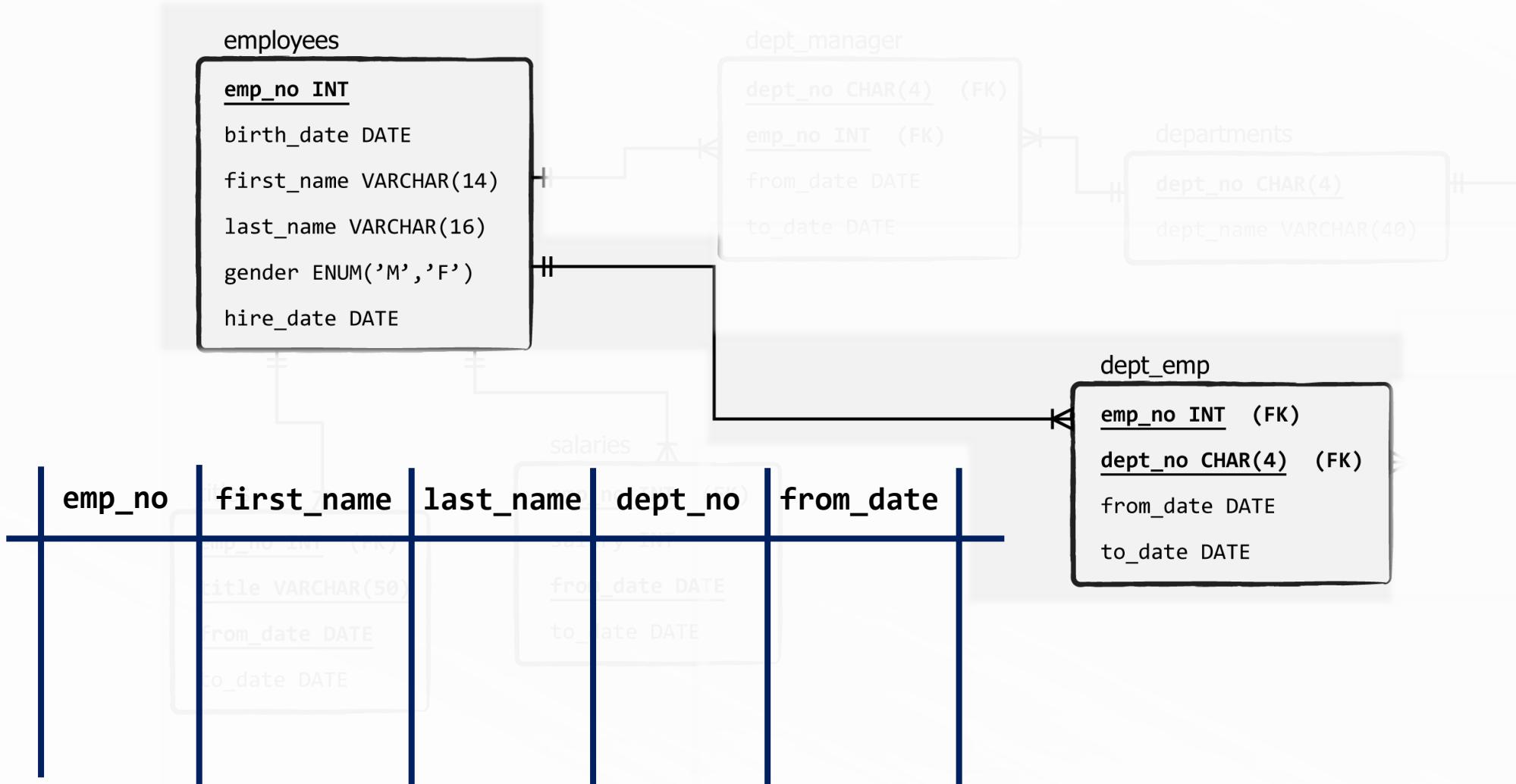
Introduction to Joins



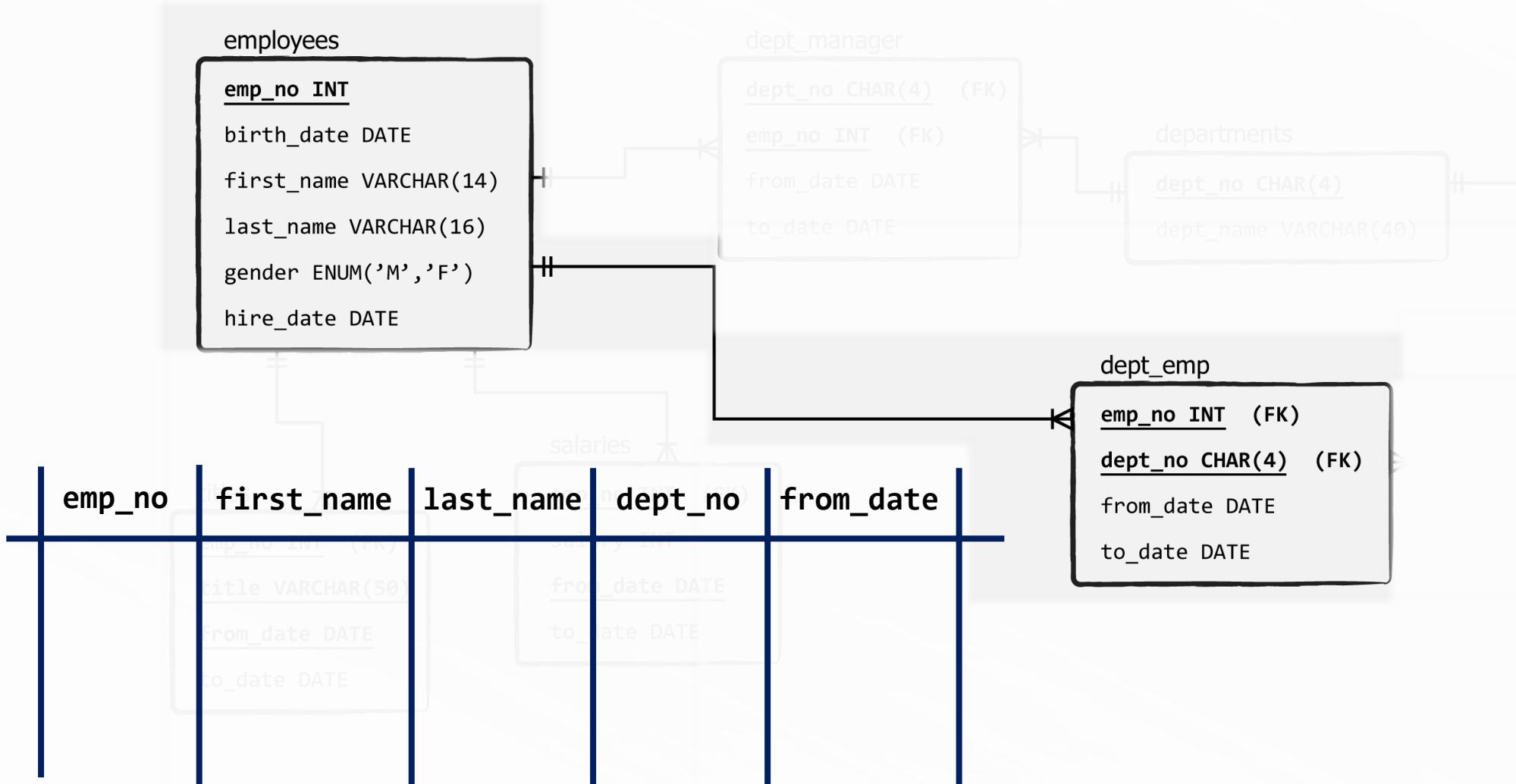
Introduction to Joins



Introduction to Joins



Introduction to Joins

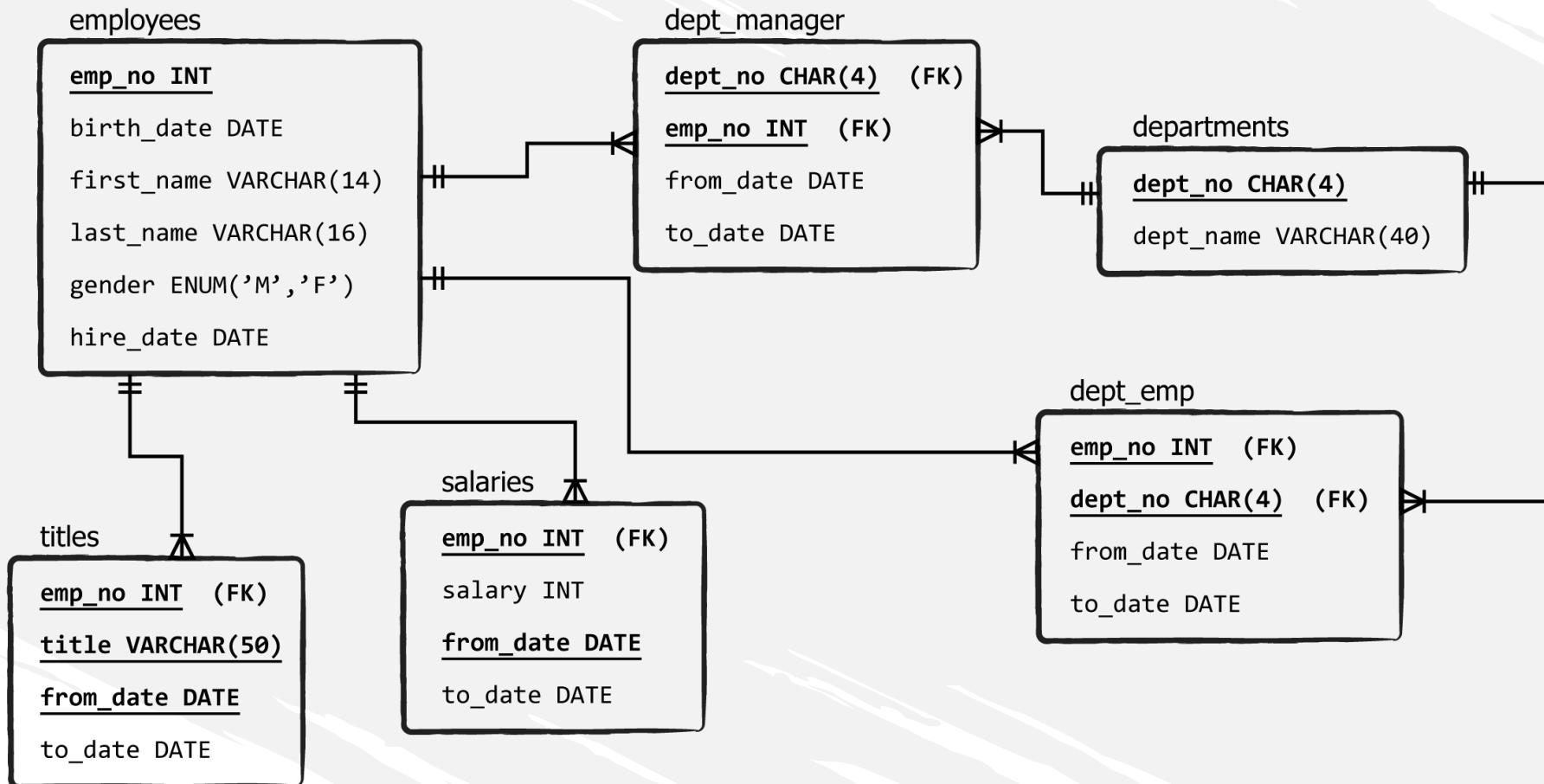


Introduction to Joins

- **joins**

- the columns you use to relate tables must represent the same object, such as *id*
- the tables you are considering need not be logically adjacent

Introduction to Joins



Introduction to Joins

- We will use two duplicate tables:

- ‘departments_dup’
- ‘dept_manager_dup’

Next:

INNER JOIN

A large conference room with rows of chairs facing a central table, set against a window overlooking a runway.

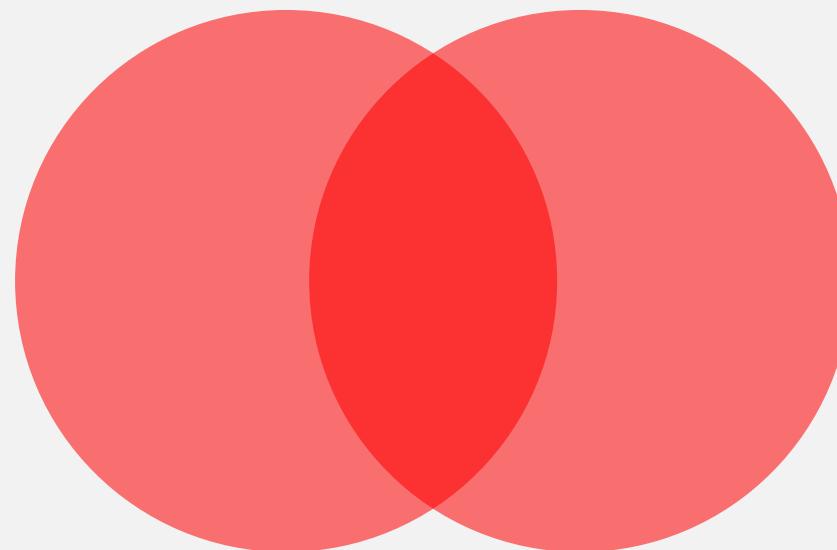
INNER JOIN

INNER JOIN

- INNER JOIN

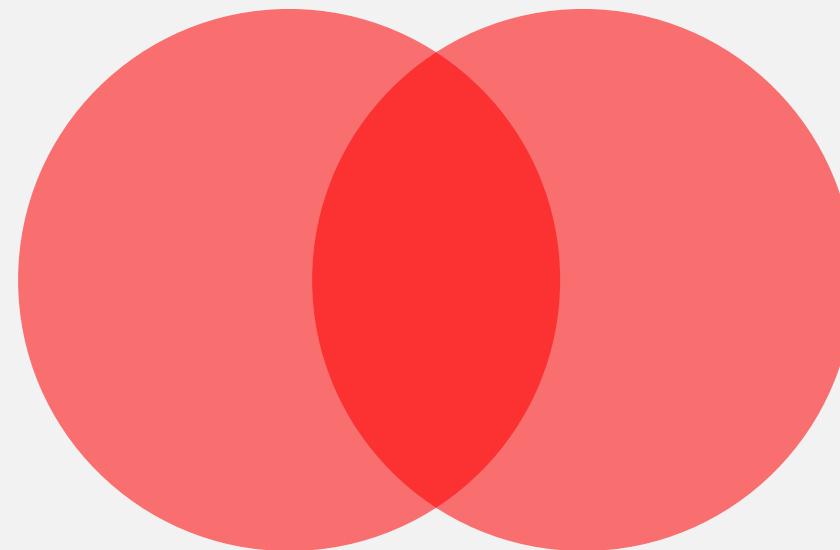
INNER JOIN

- INNER JOIN



INNER JOIN

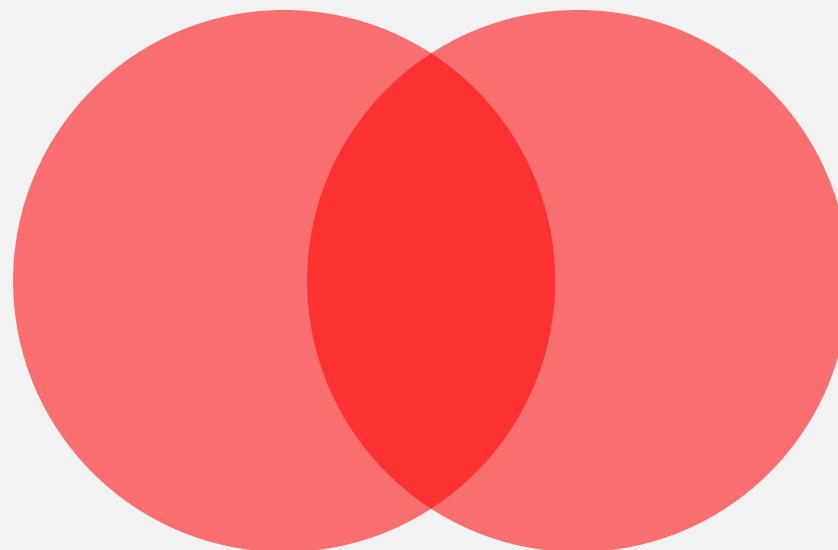
- INNER JOIN



Venn diagram

INNER JOIN

- **INNER JOIN**

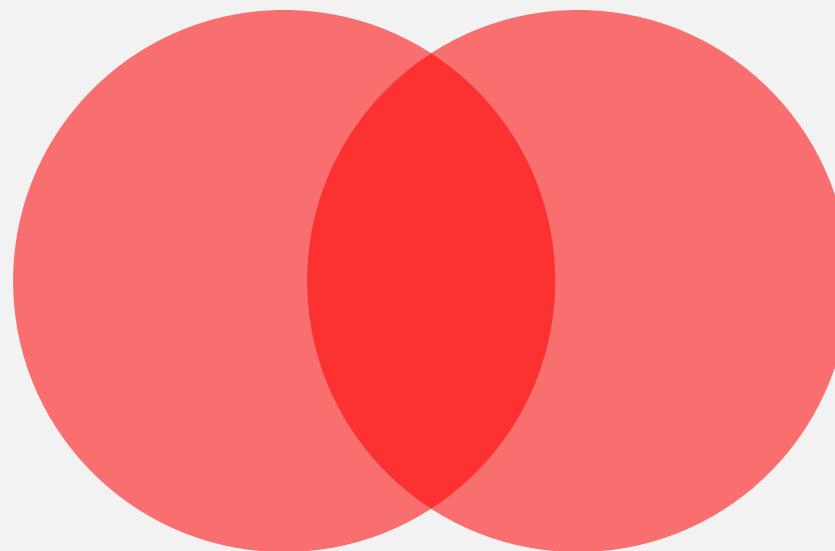


Venn diagram

a mathematical tool representing all possible logical relations
between a finite collection of sets

INNER JOIN

dept_manager_dup
dept_no CHAR(4)
emp_no INT
from_date DATE
to_date DATE



Venn diagram

a mathematical tool representing all possible logical relations
between a finite collection of sets

INNER JOIN

dept_manager_dup

dept_no CHAR(4)

emp_no INT

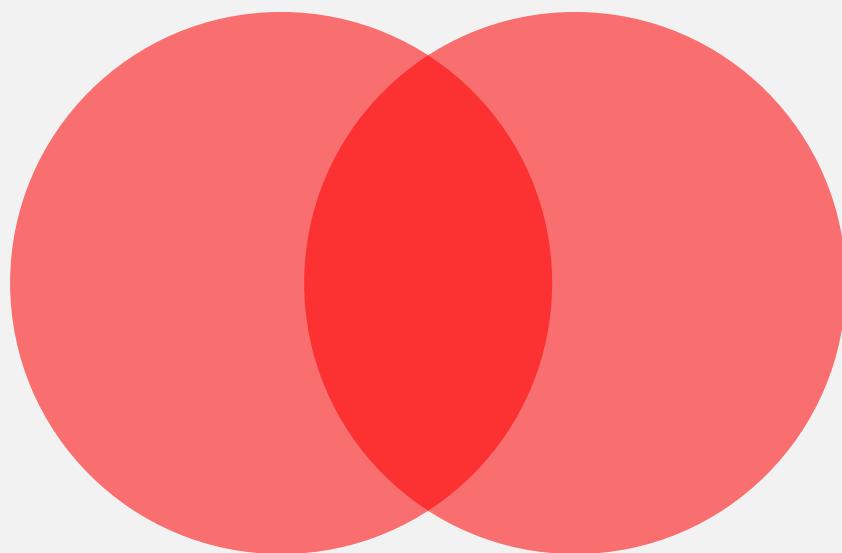
from_date DATE

to_date DATE

departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)



Venn diagram

a mathematical tool representing all possible logical relations
between a finite collection of sets

INNER JOIN

dept_manager_dup

dept_no CHAR(4)

emp_no INT

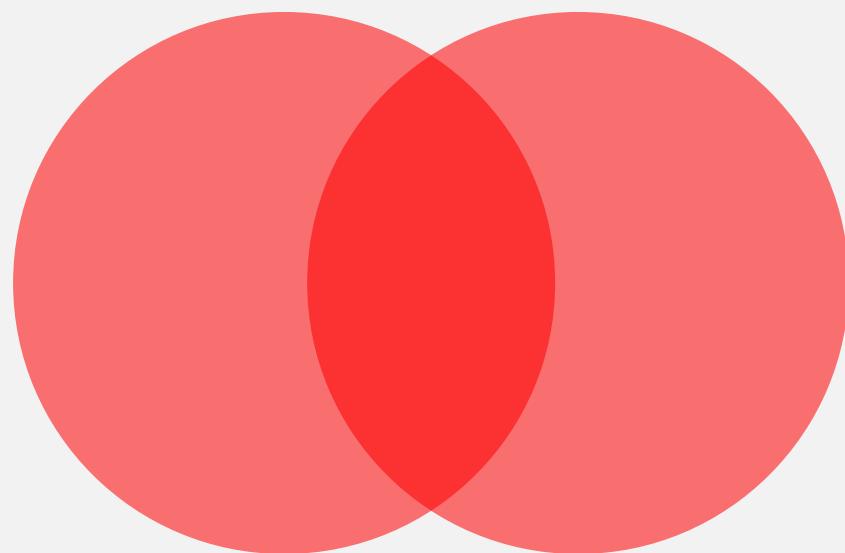
from_date DATE

to_date DATE

departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)



Which will be the related column here?

INNER JOIN

dept_manager_dup

dept_no CHAR(4)

emp_no INT

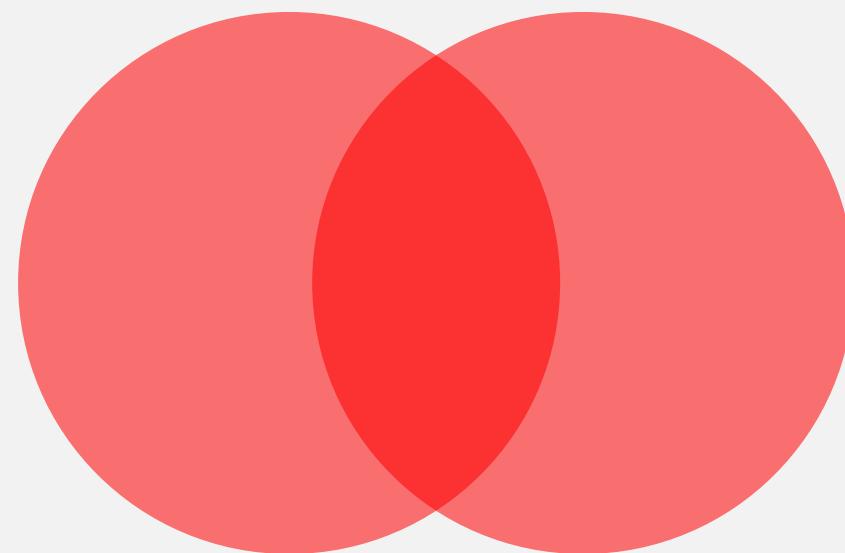
from_date DATE

to_date DATE

departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)



Related column: dept_no

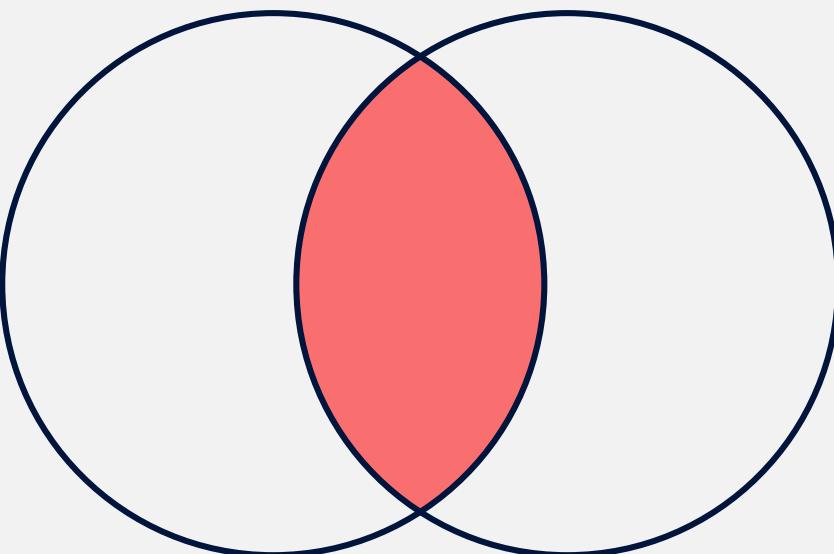
INNER JOIN

dept_manager_dup

dept_no CHAR(4)
emp_no INT
from_date DATE
to_date DATE

departments_dup

dept_no CHAR(4)
dept_name VARCHAR(40)



the area that belongs to both circles, which is filled with red, represents all records belonging to both the “Department Manager Duplicate” and the “Departments Duplicate” tables

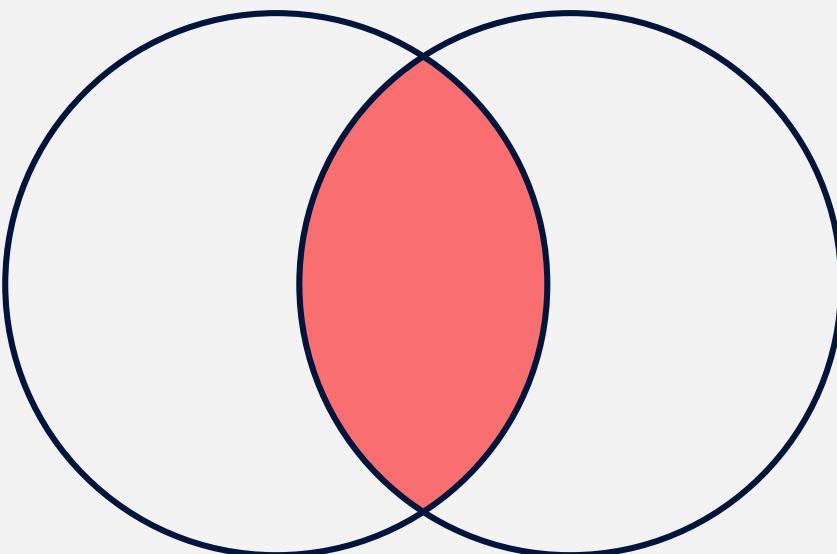
INNER JOIN

dept_manager_dup

dept_no CHAR(4)
emp_no INT
from_date DATE
to_date DATE

departments_dup

dept_no CHAR(4)
dept_name VARCHAR(40)



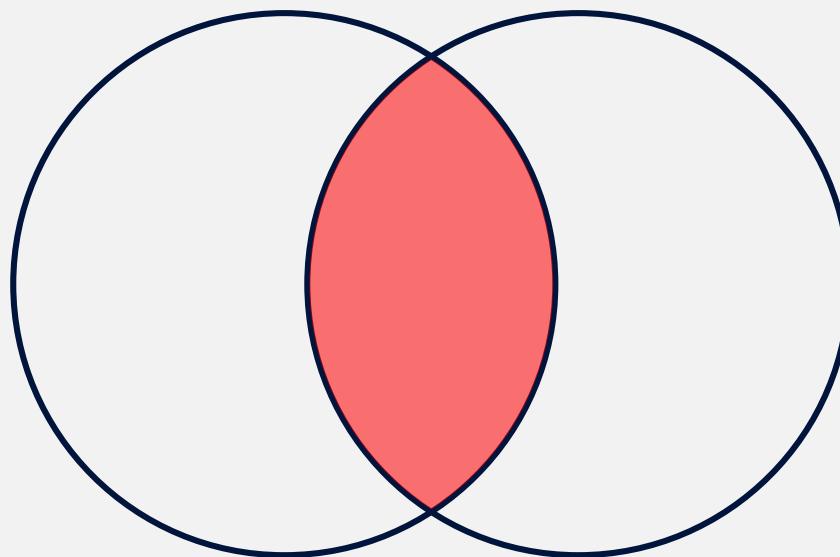
result set

the area that belongs to both circles, which is filled with red, represents all records belonging to *both* the “Department Manager Duplicate” and the “Departments Duplicate” tables

INNER JOIN

- **INNER JOIN**

can help us extract this result set

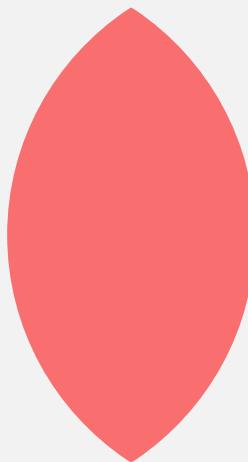


result set

INNER JOIN

- **INNER JOIN**

can help us extract this result set



result set

INNER JOIN

dept_manager_dup

dept_no CHAR(4)

emp_no INT

from_date DATE

to_date DATE

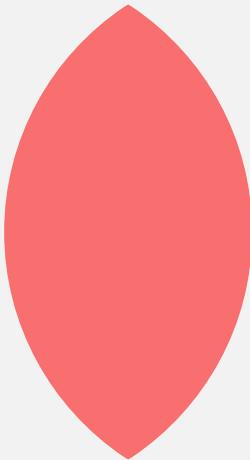
dept_no CHAR(4)

departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)

INNER JOIN



matching values = matching records

INNER JOIN

dept_manager_dup

dept_no CHAR(4)

emp_no INT

from_date DATE

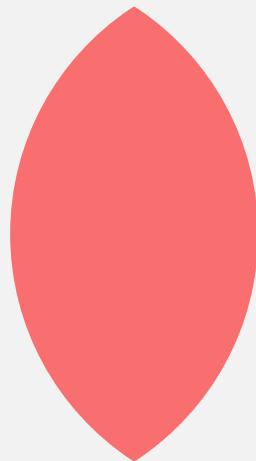
to_date DATE

dept_no CHAR(4)

departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)



matching values = matching records

INNER JOIN

dept_manager_dup

dept_no CHAR(4)

emp_no INT

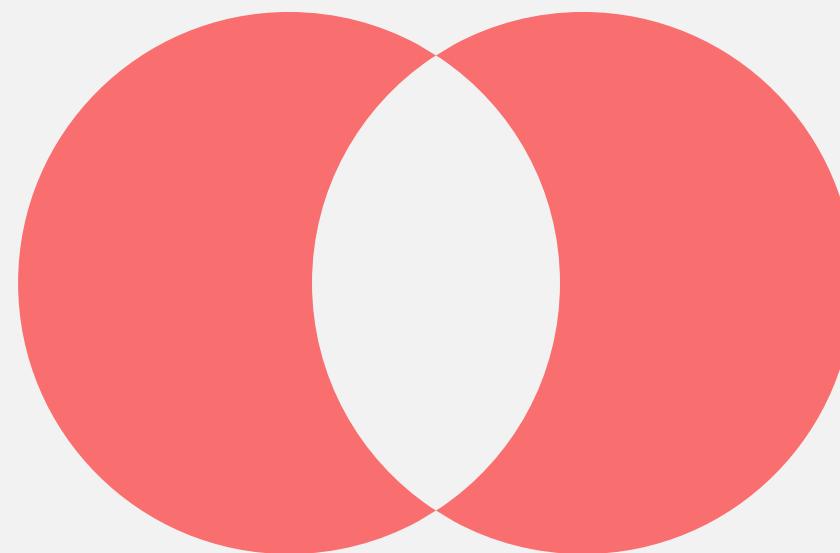
from_date DATE

to_date DATE

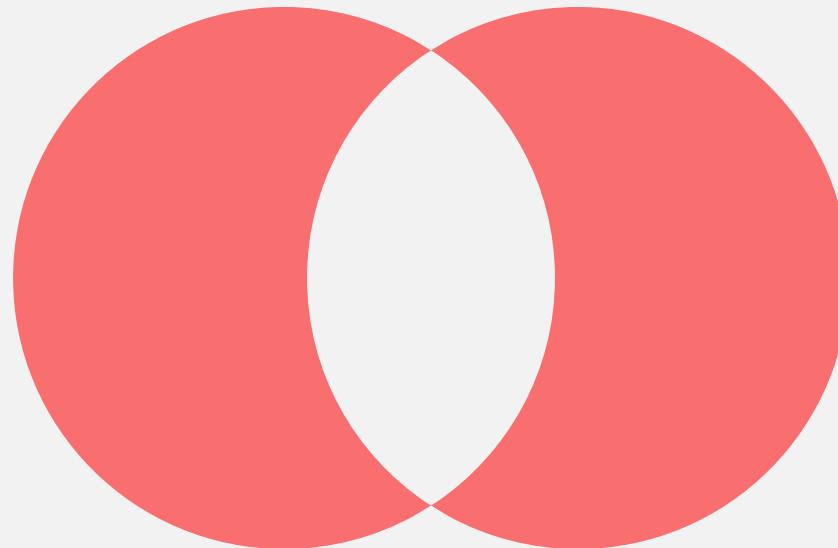
departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)



INNER JOIN



non-matching values = non-matching records

INNER JOIN

dept_manager_dup

dept_no CHAR(4)

emp_no INT

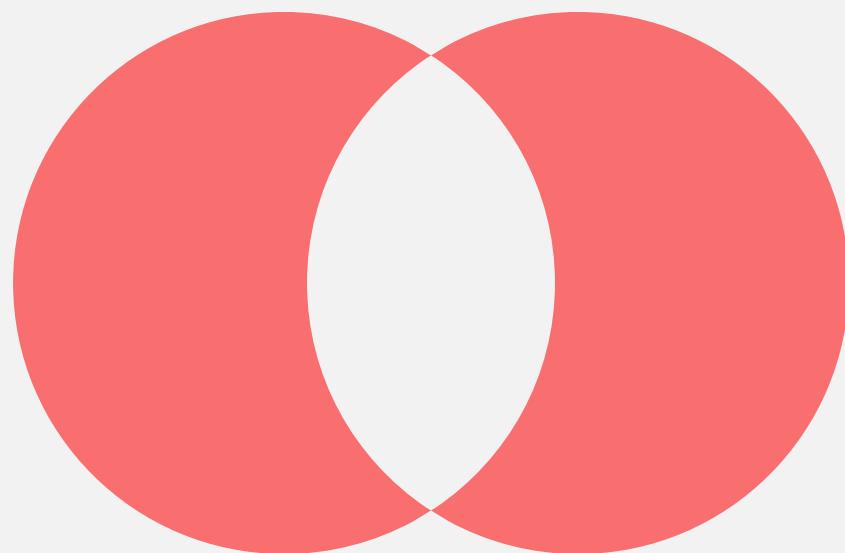
from_date DATE

to_date DATE

departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)



non-matching values = non-matching records

INNER JOIN

INNER JOIN



SQL

```
SELECT
    table_1.column_name(s), table_2.column_name(s)
FROM
    table_1
JOIN
    table_2 ON table_1.column_name = table_2.column_name;
```

INNER JOIN

INNER JOIN



SQL

```
SELECT
    t1.column_name, t1.column_name, ..., t2.column_name, ...
FROM
    table_1 t1
JOIN
    table_2 t2 ON t1.column_name = t2.column_name;
```

INNER JOIN

dept_manager_dup

dept_no CHAR(4)

emp_no INT

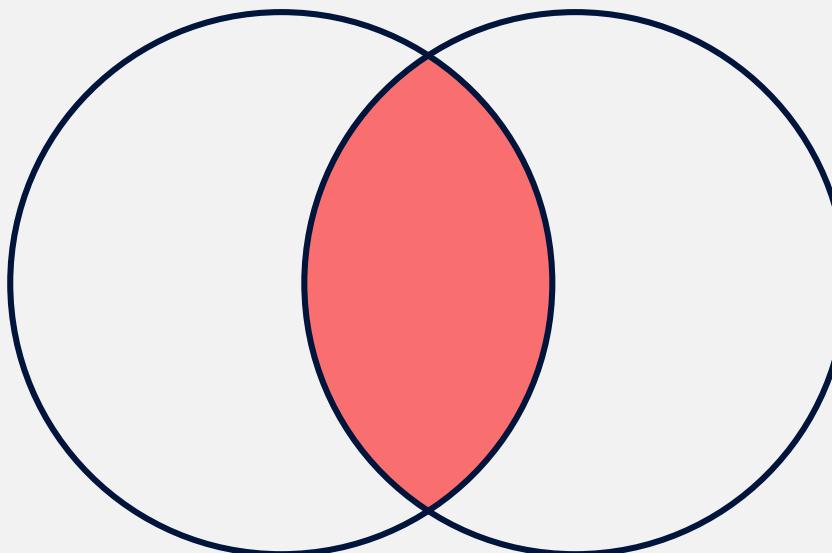
from_date DATE

to_date DATE

departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)



INNER JOIN

M

dept_manager_dup

dept_no CHAR(4)

emp_no INT

from_date DATE

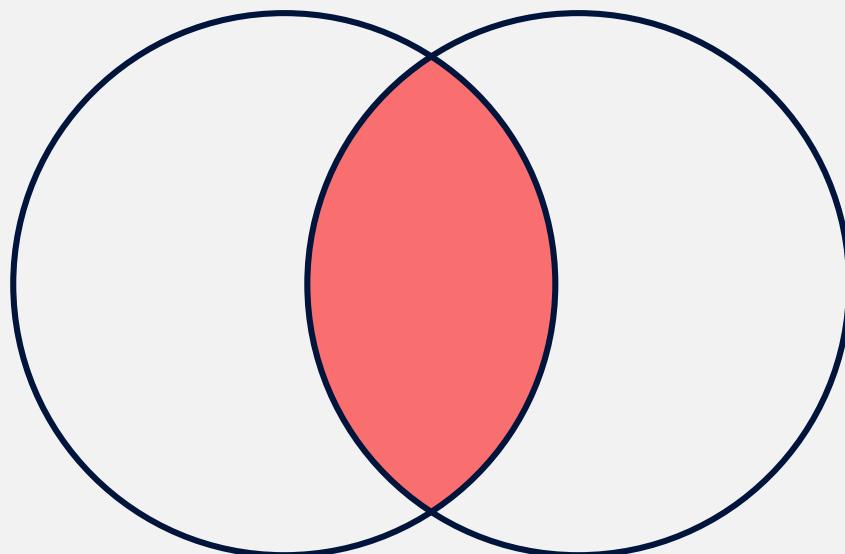
to_date DATE

D

departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)



INNER JOIN

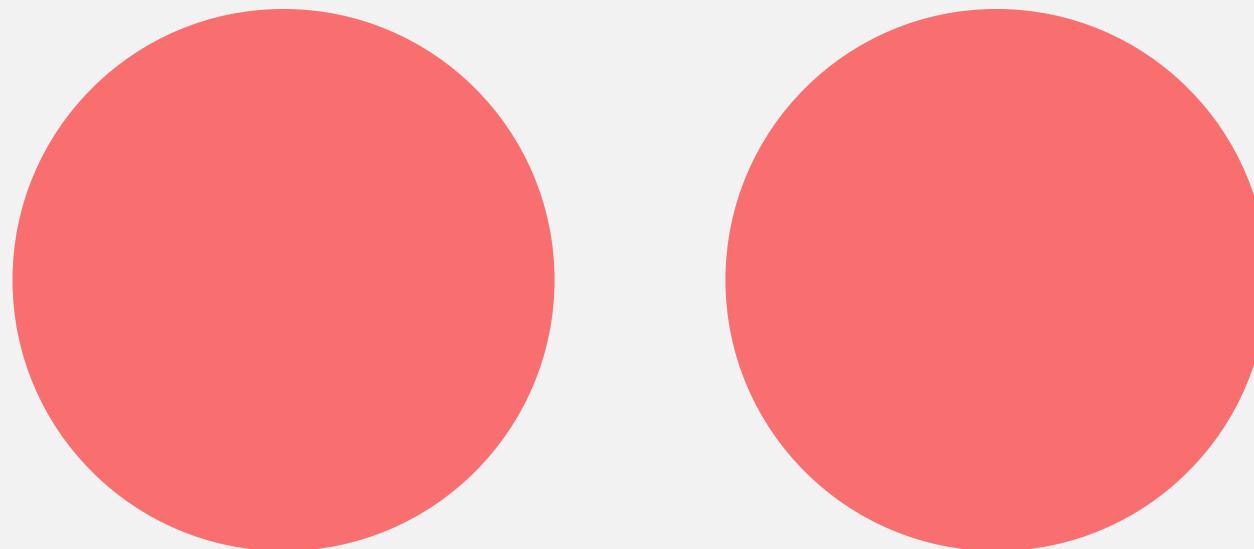
- inner joins extract only records in which the values in the related columns match. Null values, or values appearing in just one of the two tables and not appearing in the other, are not displayed
 - only non-null matching values are in play

INNER JOIN

- And what if such *matching values* did not exist?

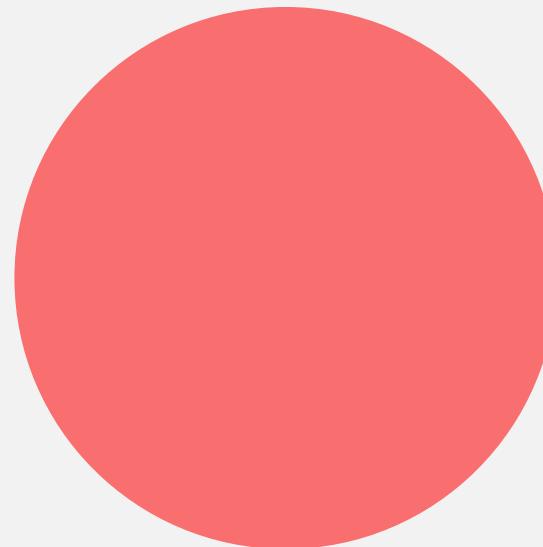
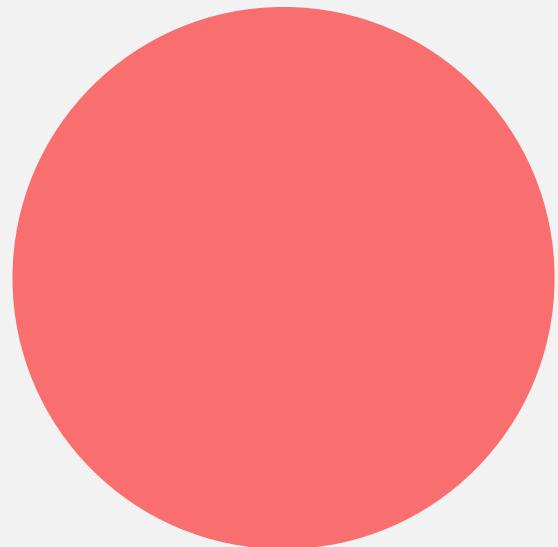
INNER JOIN

- And what if such *matching values* did not exist?



INNER JOIN

- And what if such *matching values* did not exist?



Simply, the result set will be empty. There will be no link between the two tables.

A large, modern conference room with a long rectangular table in the center. Rows of black office chairs are arranged facing the table. The room has a high ceiling with a grid of recessed lights. Large windows on one side provide a view of an outdoor area.

Duplicate Records

Duplicate Records

- duplicate records, also known as duplicate rows, are identical rows in an SQL table

Duplicate Records

- duplicate records, also known as duplicate rows, are identical rows in an SQL table
 - for a pair of duplicate records, the values in each column coincide

Duplicate Records

- duplicate rows are not always allowed in a database or a data table

Duplicate Records

- duplicate rows are not always allowed in a database or a data table
 - they are sometimes encountered, especially in new, raw, or uncontrolled data

Duplicate Records

- duplicate rows are not always allowed in a database or a data table
 - they are sometimes encountered, especially in new, raw, or uncontrolled data

here's how to handle duplicates:

GROUP BY the field that differs most among records!

A large conference room with rows of chairs facing a central table. The room has a modern design with a glass partition and a large window overlooking a landscape.

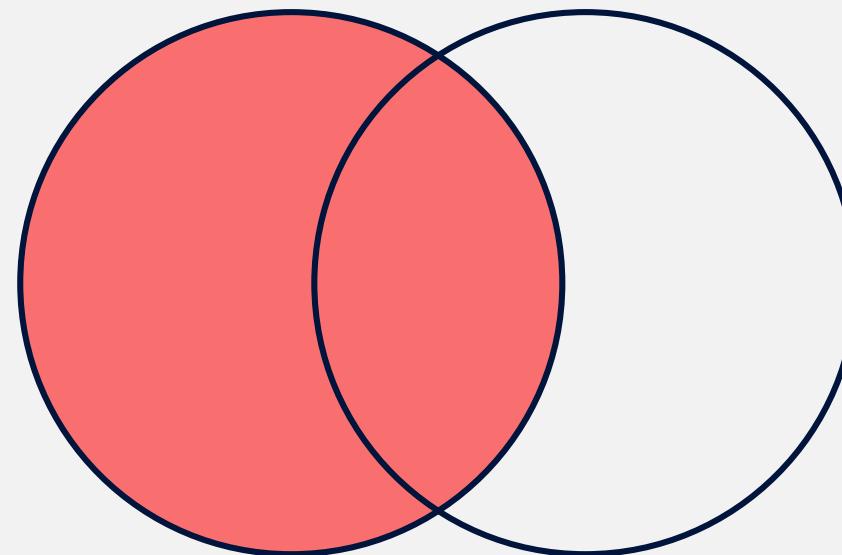
LEFT JOIN

LEFT JOIN

- LEFT JOIN

LEFT JOIN

- **LEFT JOIN**



LEFT JOIN

dept_manager_dup

dept_no CHAR(4)

emp_no INT

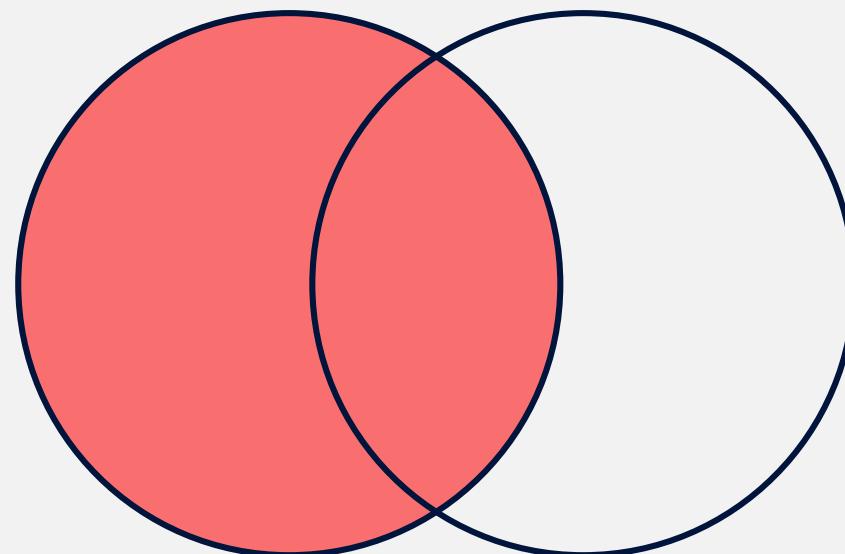
from_date DATE

to_date DATE

departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)



LEFT JOIN

dept_manager_dup

dept_no CHAR(4)

emp_no INT

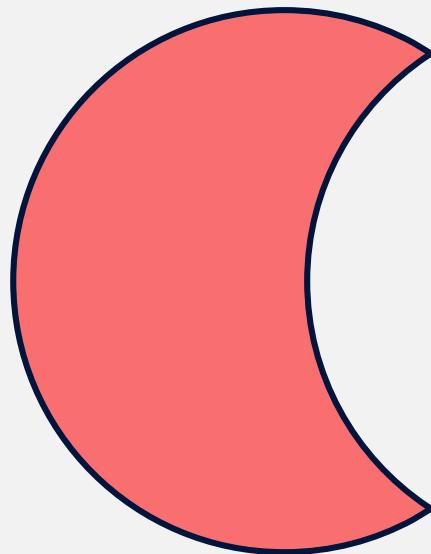
from_date DATE

to_date DATE

departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)



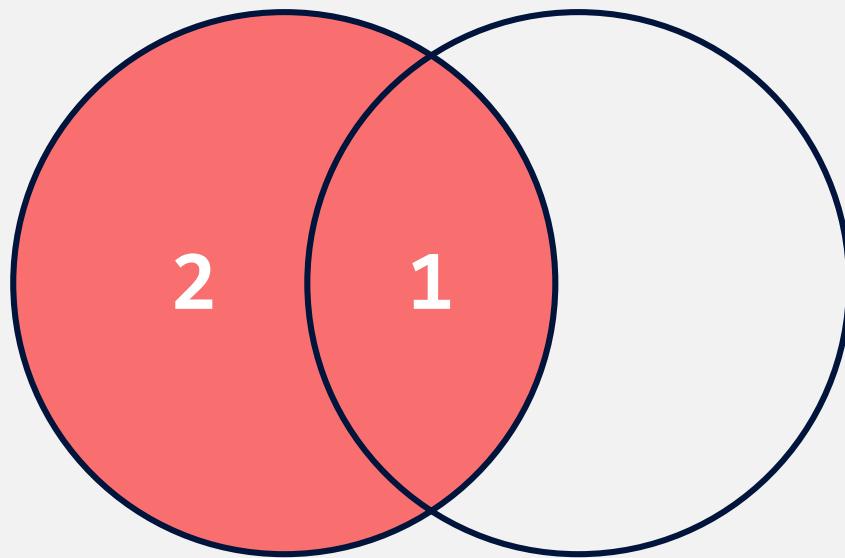
LEFT JOIN

dept_manager_dup

dept_no CHAR(4)
emp_no INT
from_date DATE
to_date DATE

departments_dup

dept_no CHAR(4)
dept_name VARCHAR(40)



- 1) all matching values of the two tables +
- 2) all values from the left table that match no values from the right table

LEFT JOIN

dept_manager_dup

dept_no CHAR(4)

emp_no INT

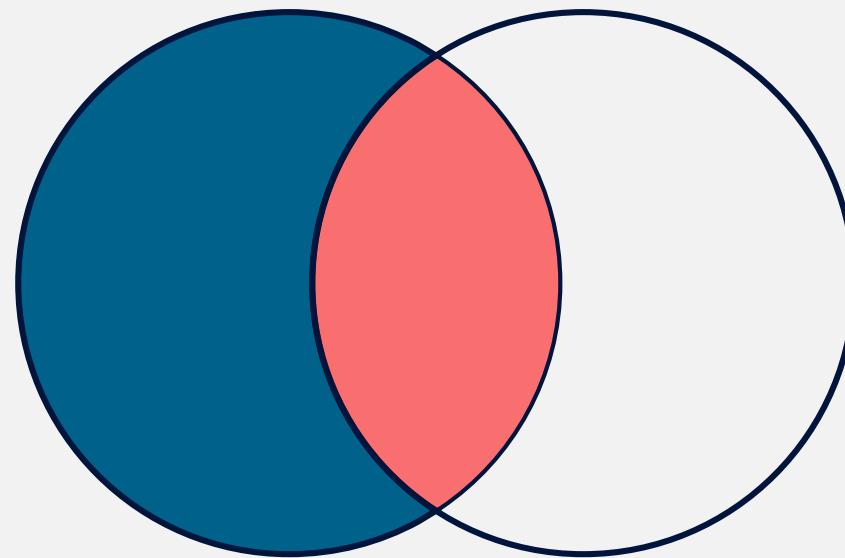
from_date DATE

to_date DATE

departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)



all matching values of the two tables +
all values from the left table that match no values from the right table

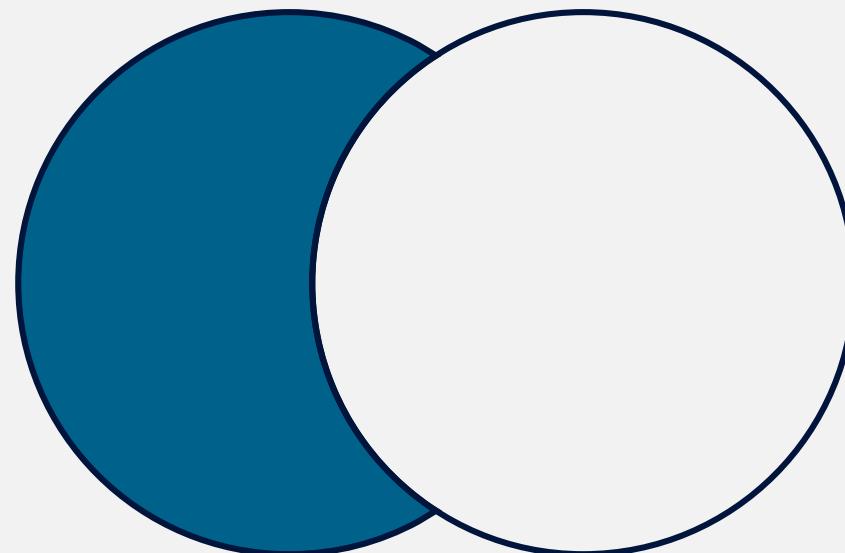
LEFT JOIN

dept_manager_dup

dept_no CHAR(4)
emp_no INT
from_date DATE
to_date DATE

departments_dup

dept_no CHAR(4)
dept_name VARCHAR(40)



~~all matching values of the two tables +~~

all values from the left table that match no values from the right table

LEFT JOIN

dept_manager_dup

dept_no CHAR(4)

emp_no INT

from_date DATE

to_date DATE

departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)

~~all matching values of the two tables +~~

all values from the left table that match no values from the right table

LEFT JOIN

- **LEFT JOIN**

when working with left joins, the order in which you join tables *matters*

LEFT JOIN

dept_manager_dup

dept_no CHAR(4)

emp_no INT

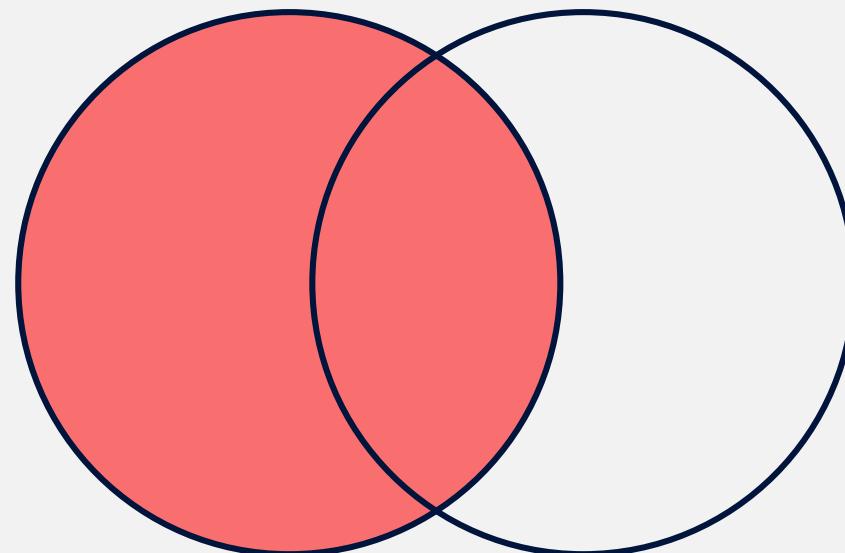
from_date DATE

to_date DATE

departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)

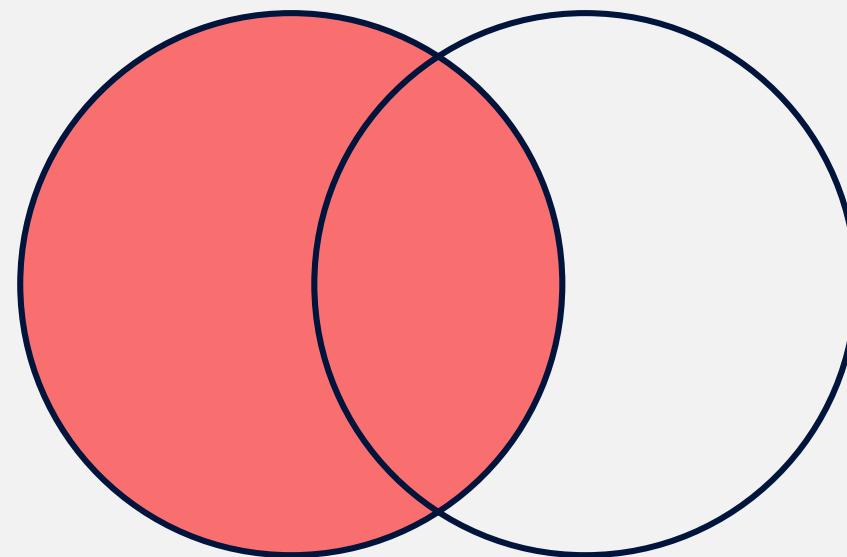


LEFT JOIN

departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)



dept_manager_dup

dept_no CHAR(4)

emp_no INT

from_date DATE

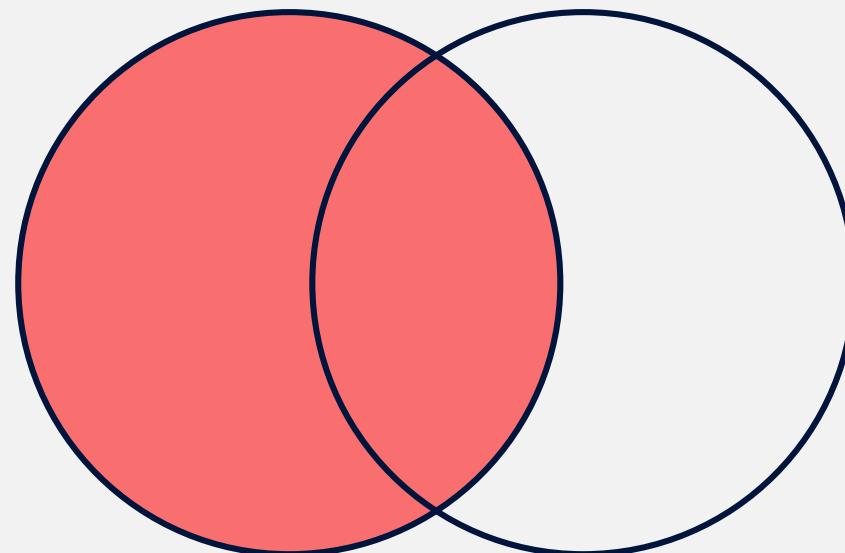
to_date DATE

departments_dup

```
dept_no CHAR(4)  
dept_name VARCHAR(40)
```

dept_no	dept_name
NULL	Public Relations
d001	Marketing
d003	Human Resources
d004	Production
d005	Development
d006	Quality Management
d007	Sales
d008	Research
d009	Customer Service
d010	NULL
d011	NULL

LEFT JOIN

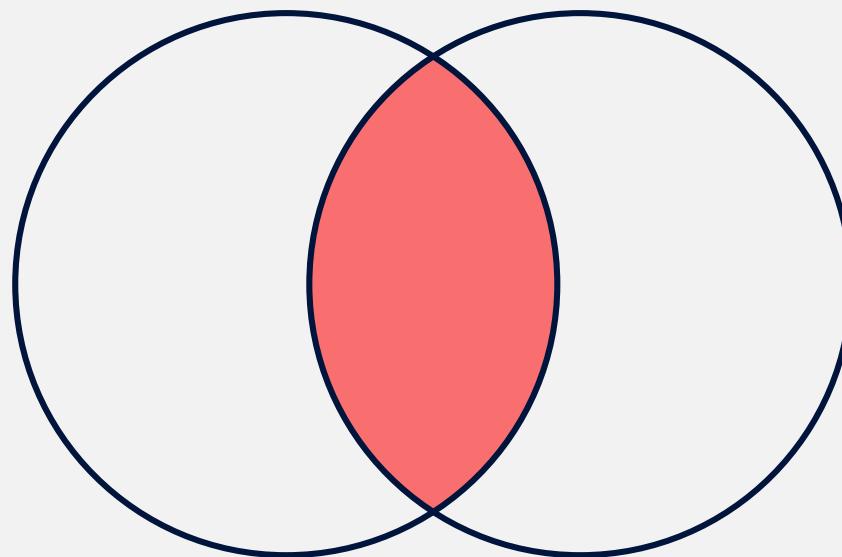


dept_manager_dup

```
dept_no CHAR(4)  
emp_no INT  
from_date DATE  
to_date DATE
```

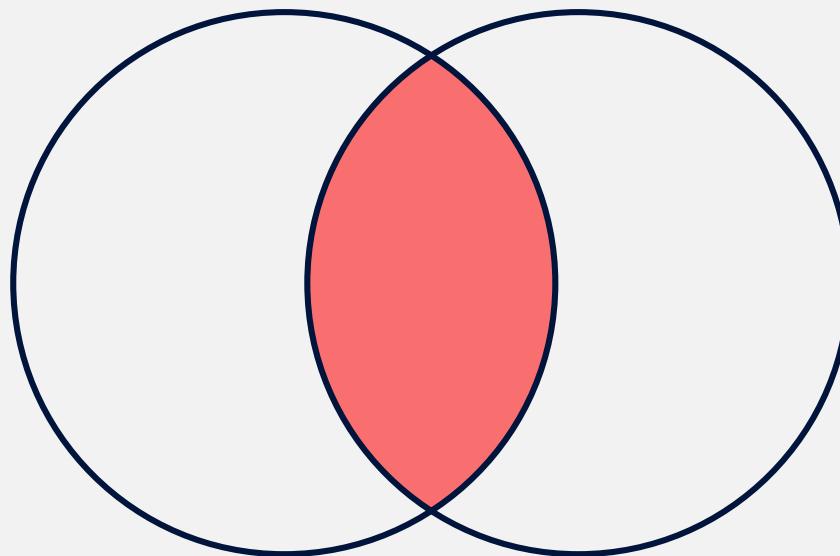
emp_no	dept_no	from_date	to_date
999904	NULL	2017-01-01	NULL
999905	NULL	2017-01-01	NULL
999906	NULL	2017-01-01	NULL
999907	NULL	2017-01-01	NULL
110085	d002	1985-01-01	1989-12-17
110114	d002	1989-12-17	9999-01-01
110183	d003	1985-01-01	1992-03-21
110228	d003	1992-03-21	9999-01-01
110303	d004	1985-01-01	1988-09-09
110344	d004	1988-09-09	1992-08-02
110386	d004	1992-08-02	1996-08-30
110420	d004	1996-08-30	9999-01-01
110511	d005	1985-01-01	1992-04-25
110567	d005	1992-04-25	9999-01-01
110725	d006	1985-01-01	1989-05-06
110765	d006	1989-05-06	1991-09-12
110800	d006	1991-09-12	1994-06-28
110854	d006	1994-06-28	9999-01-01
111035	d007	1985-01-01	1991-03-07
111133	d007	1991-03-07	9999-01-01
111400	d008	1985-01-01	1991-04-08
111534	d008	1991-04-08	9999-01-01
111692	d009	1985-01-01	1988-10-17
111784	d009	1988-10-17	1992-09-08
111877	d009	1992	
111939	d009	1996	

LEFT JOIN



INNER join

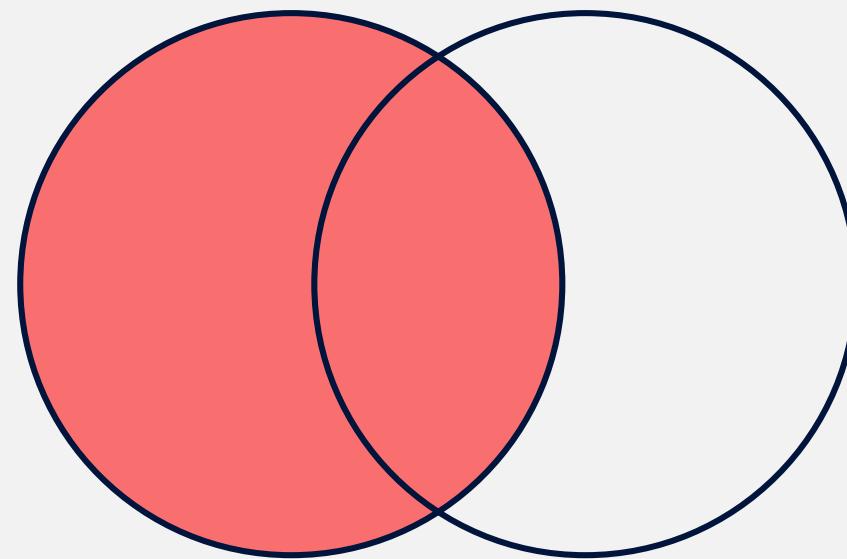
LEFT JOIN



INNER join

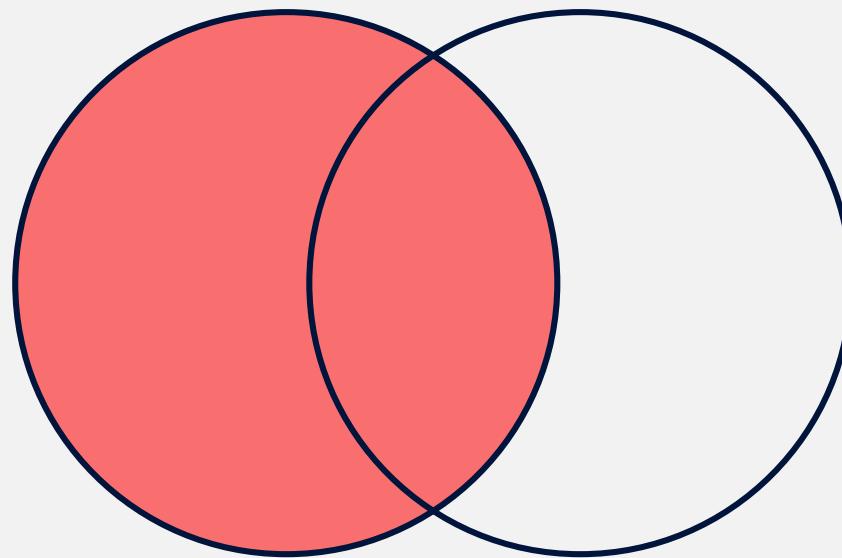
the result set is in the *inner* part of the Venn diagram

LEFT JOIN



LEFT join

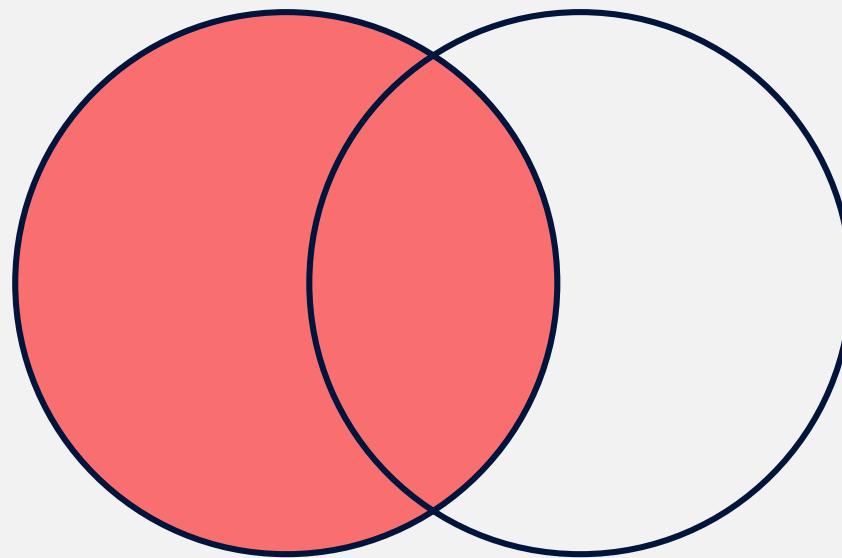
LEFT JOIN



LEFT join

in the output obtained you have data from the
outer part of the Venn diagram too

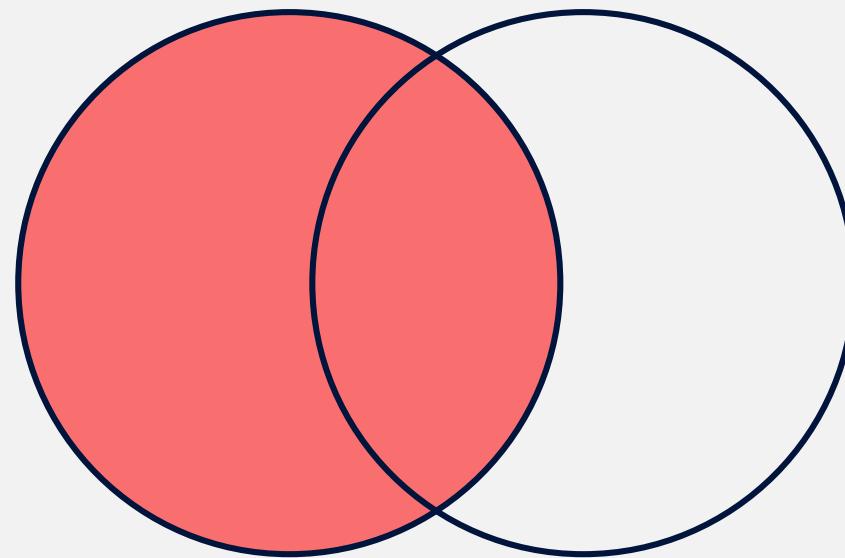
LEFT JOIN



LEFT join = LEFT OUTER join

in the output obtained you have data from the
outer part of the Venn diagram too

LEFT JOIN



LEFT join = LEFT OUTER join

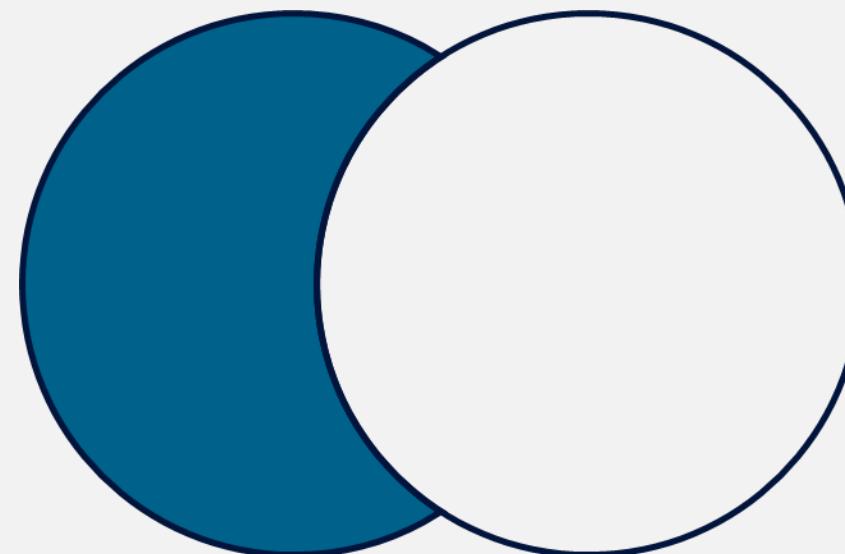
if you are using a left join, it will always be an
OUTER type of join

LEFT JOIN

- left joins can deliver a list with all records from the left table that do not match any rows from the right table

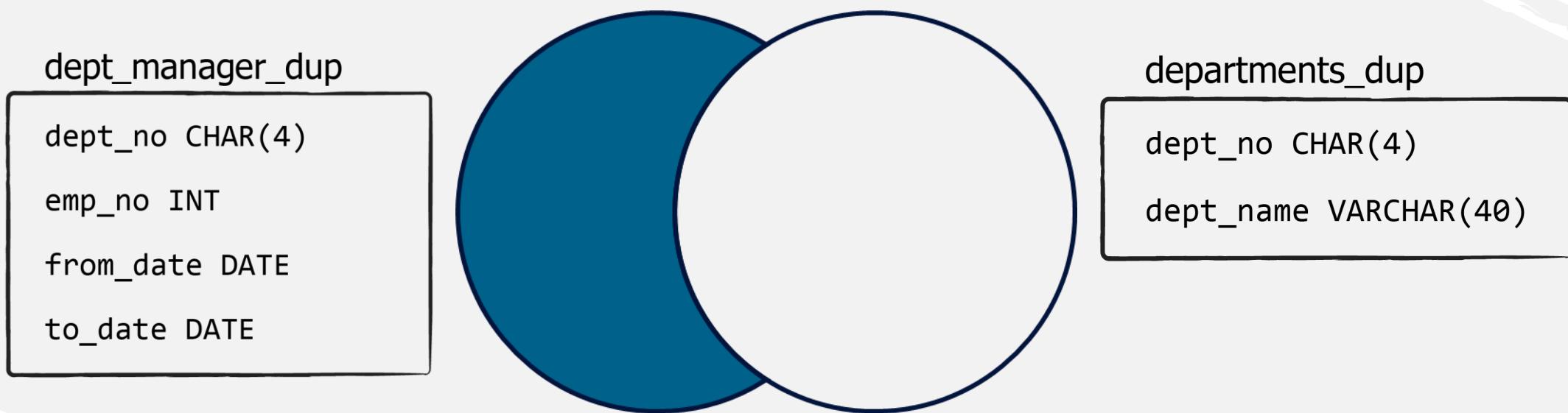
LEFT JOIN

- left joins can deliver a list with all records from the left table that do not match any rows from the right table



LEFT JOIN

left joins can deliver a list with all records from the left table that do not match any rows from the right table



LEFT JOIN

dept_manager_dup

dept_no CHAR(4)

emp_no INT

from_date DATE

to_date DATE

departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)



SQL

```
SELECT
    t1.column_name, t1.column_name, ..., t2.column_name, ...
FROM
    table_1 t1
JOIN
    table_2 t2 ON t1.column_name = t2.column_name
WHERE
    column_name ... IS NULL;
```

A large conference room with rows of black office chairs facing a long rectangular table. The room has a modern design with a grid-patterned ceiling and floor-to-ceiling windows on one side, offering a view of an airport tarmac.

RIGHT JOIN

RIGHT JOIN

- **RIGHT JOIN**

RIGHT JOIN

- **RIGHT JOIN**

their functionality is identical to LEFT JOINS, with the only difference being that the direction of the operation is inverted

RIGHT JOIN

- **RIGHT JOIN** = **RIGHT OUTER JOIN**

their functionality is identical to LEFT JOINS, with the only difference being that the direction of the operation is inverted

RIGHT JOIN

dept_manager_dup

dept_no CHAR(4)

emp_no INT

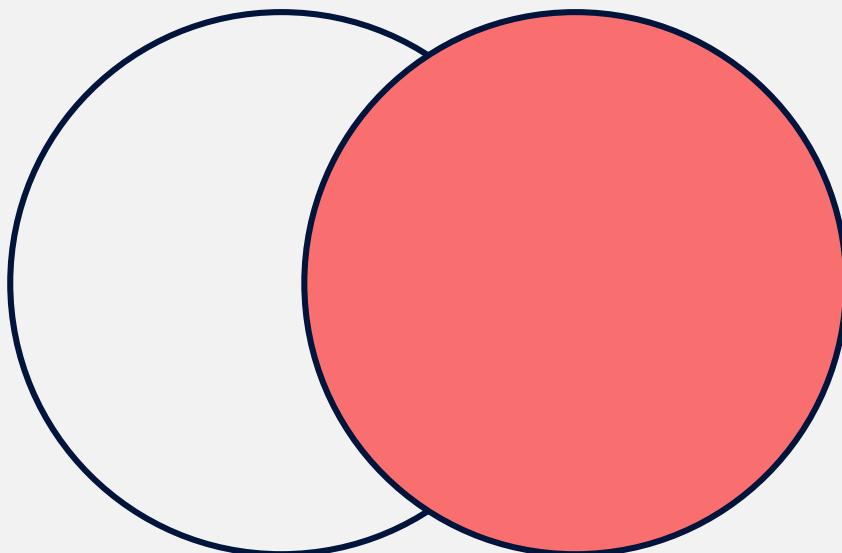
from_date DATE

to_date DATE

departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)



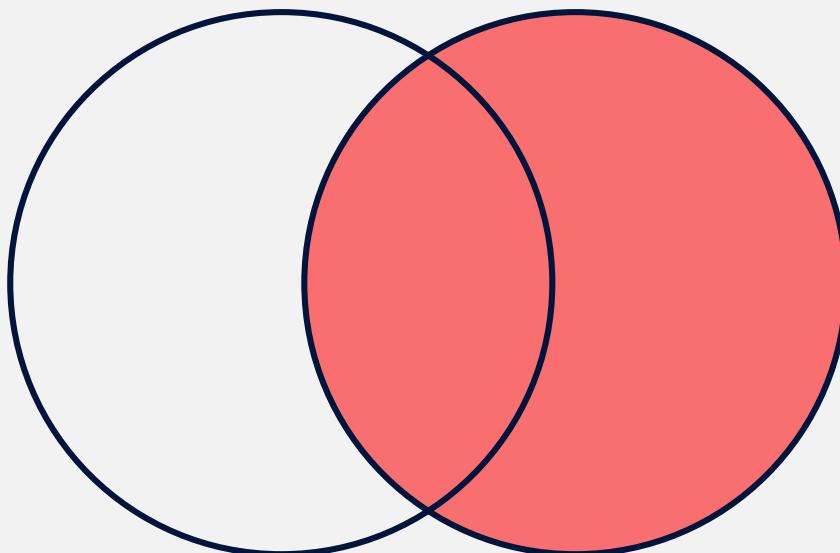
RIGHT JOIN

dept_manager_dup

dept_no CHAR(4)
emp_no INT
from_date DATE
to_date DATE

departments_dup

dept_no CHAR(4)
dept_name VARCHAR(40)



RIGHT JOIN

dept_manager_dup

dept_no CHAR(4)

emp_no INT

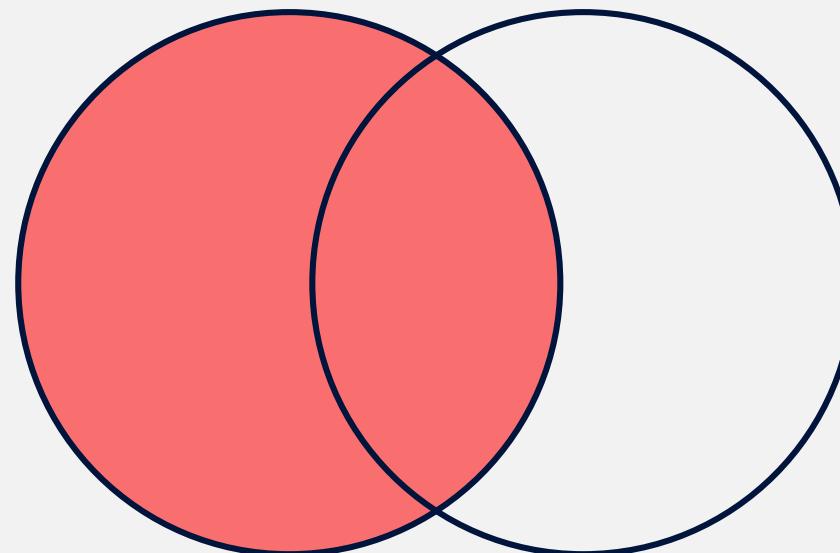
from_date DATE

to_date DATE

departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)



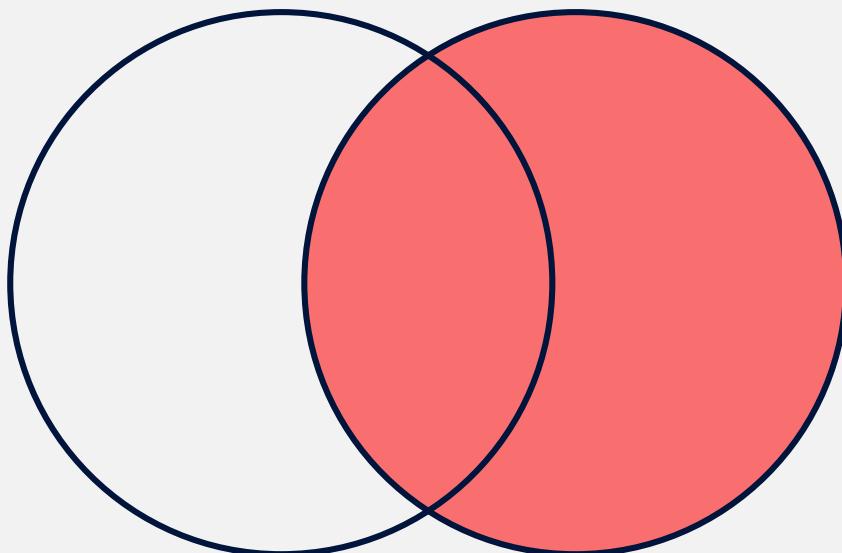
RIGHT JOIN

dept_manager_dup

dept_no CHAR(4)
emp_no INT
from_date DATE
to_date DATE

departments_dup

dept_no CHAR(4)
dept_name VARCHAR(40)



RIGHT JOIN

dept_manager_dup

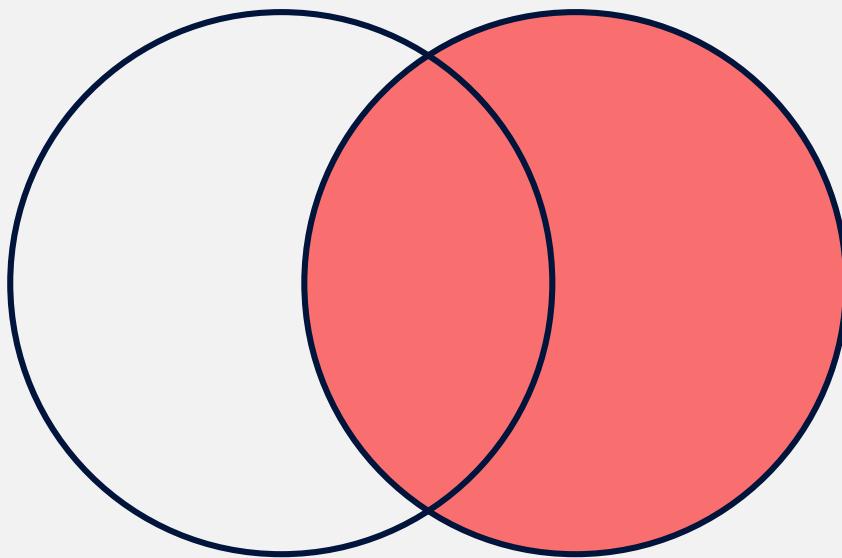
```
dept_no CHAR(4)  
emp_no INT  
from_date DATE  
to_date DATE
```

emp_no	dept_no	from_date	to_date
999904	NULL	2017-01-01	NULL
999905	NULL	2017-01-01	NULL
999906	NULL	2017-01-01	NULL
999907	NULL	2017-01-01	NULL
110085	d002	1985-01-01	1989-12-17
110114	d002	1989-12-17	9999-01-01
110183	d003	1985-01-01	1992-03-21
110228	d003	1992-03-21	9999-01-01
110303	d004	1985-01-01	1988-09-09
110344	d004	1988-09-09	1992-08-02
110386	d004	1992-08-02	1996-08-30
110420	d004	1996-08-30	9999-01-01
110511	d005	1985-01-01	1992-04-25
110567	d005	1992-04-25	9999-01-01
110725	d006	1985-01-01	1989-05-06
110765	d006	1989-05-06	1991-09-12
110800	d006	1991-09-12	1994-06-28
110854	d006	1994-06-28	9999-01-01
111035	d007	1985-01-01	1991-03-07
111133	d007	1991-03-07	9999-01-01
111400	d008	1985-01-01	1991-04-08
111534	d008	1991-04-08	9999-01-01
111692	d009	1985-01-01	1988-10-17
111784	d009	1988-10-17	1992-09-08
111877	d009	1992-09-08	1996-01-03
111939	d009	1996-01-03	9999-01-01

departments_dup

```
dept_no CHAR(4)  
dept_name VARCHAR(40)
```

dept_no	dept_name
NULL	Public Relations
d001	Marketing
d003	Human Resources
d004	Production
d005	Development
d006	Quality Management
d007	Sales
d008	Research
d009	Customer Service
d010	NULL
d011	NULL



RIGHT JOIN

- Whether we run a RIGHT JOIN or a LEFT JOIN with an *inverted tables order*, we will obtain the same output, right?

RIGHT JOIN

- Whether we run a RIGHT JOIN or a LEFT JOIN with an *inverted* tables order, we will obtain the same output, right?

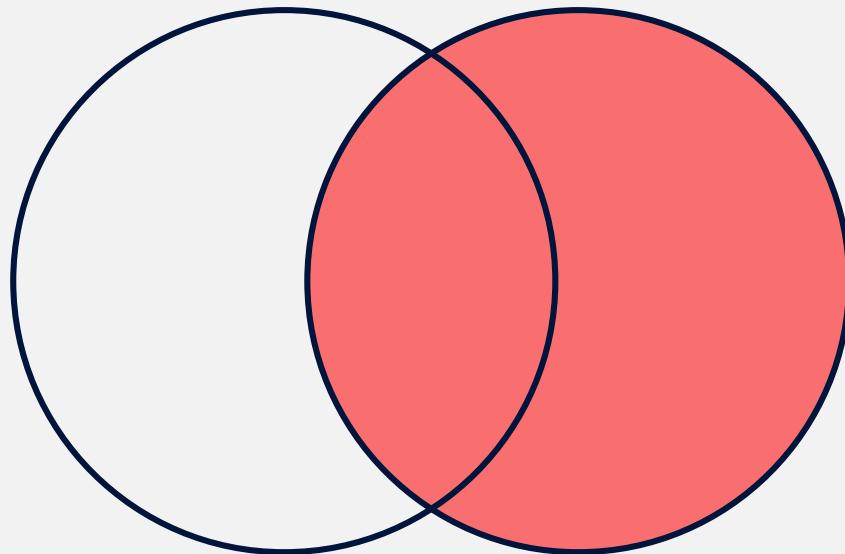
Yes, we will!

departments_dup

```
dept_no CHAR(4)  
dept_name VARCHAR(40)
```

dept_no	dept_name
NULL	Public Relations
d001	Marketing
d003	Human Resources
d004	Production
d005	Development
d006	Quality Management
d007	Sales
d008	Research
d009	Customer Service
d010	NULL
d011	NULL

RIGHT JOIN



dept_manager_dup

```
dept_no CHAR(4)  
emp_no INT  
from_date DATE  
to_date DATE
```

emp_no	dept_no	from_date	to_date
999904	NULL	2017-01-01	NULL
999905	NULL	2017-01-01	NULL
999906	NULL	2017-01-01	NULL
999907	NULL	2017-01-01	NULL
110085	d002	1985-01-01	1989-12-17
110114	d002	1989-12-17	9999-01-01
110183	d003	1985-01-01	1992-03-21
110228	d003	1992-03-21	9999-01-01
110303	d004	1985-01-01	1988-09-09
110344	d004	1988-09-09	1992-08-02
110386	d004	1992-08-02	1996-08-30
110420	d004	1996-08-30	9999-01-01
110511	d005	1985-01-01	1992-04-25
110567	d005	1992-04-25	9999-01-01
110725	d006	1985-01-01	1989-05-06
110765	d006	1989-05-06	1991-09-12
110800	d006	1991-09-12	1994-06-28
110854	d006	1994-06-28	9999-01-01
111035	d007	1985-01-01	1991-03-07
111133	d007	1991-03-07	9999-01-01
111400	d008	1985-01-01	1991-04-08
111534	d008	1991-04-08	9999-01-01
111692	d009	1985-01-01	1988-10-17
111784	d009	1988-10-17	1992-09-08
111877	d009	1992	
111939	d009	1996	

dept_manager_dup

dept_no CHAR(4)

emp_no INT

from_date DATE

to_date DATE

emp_no	dept_no	from_date	to_date
999904	NULL	2017-01-01	NULL
999905	NULL	2017-01-01	NULL
999906	NULL	2017-01-01	NULL
999907	NULL	2017-01-01	NULL
110085	d002	1985-01-01	1989-12-17
110114	d002	1989-12-17	9999-01-01
110183	d003	1985-01-01	1992-03-21
110228	d003	1992-03-21	9999-01-01
110303	d004	1985-01-01	1988-09-09
110344	d004	1988-09-09	1992-08-02
110386	d004	1992-08-02	1996-08-30
110420	d004	1996-08-30	9999-01-01
110511	d005	1985-01-01	1992-04-25
110567	d005	1992-04-25	9999-01-01
110725	d006	1985-01-01	1989-05-06
110765	d006	1989-05-06	1991-09-12
110800	d006	1991-09-12	1994-06-28
110854	d006	1994-06-28	9999-01-01
111035	d007	1985-01-01	1991-03-07
111133	d007	1991-03-07	9999-01-01
111400	d008	1985-01-01	1991-04-08
111534	d008	1991-04-08	9999-01-01
111692	d009	1985-01-01	1988-10-17
111784	d009	1988-10-17	1992-09-08
111877	d009	1992-09-08	1996-01-03
111939	d009	1996-01-03	9999-01-01

RIGHT JOIN

departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)

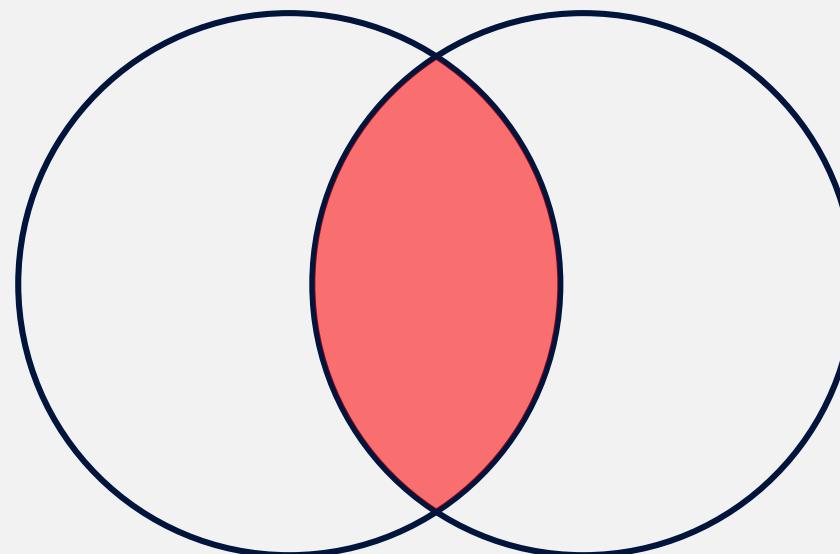
dept_no	dept_name
NULL	Public Relations
d001	Marketing
d003	Human Resources
d004	Production
d005	Development
d006	Quality Management
d007	Sales
d008	Research
d009	Customer Service
d010	NULL
d011	NULL

RIGHT JOIN

dept_manager_dup

dept_no CHAR(4)
emp_no INT
from_date DATE
to_date DATE

dept_no CHAR(4)



departments_dup

dept_no CHAR(4)
dept_name VARCHAR(40)

RIGHT JOIN

dept_manager_dup

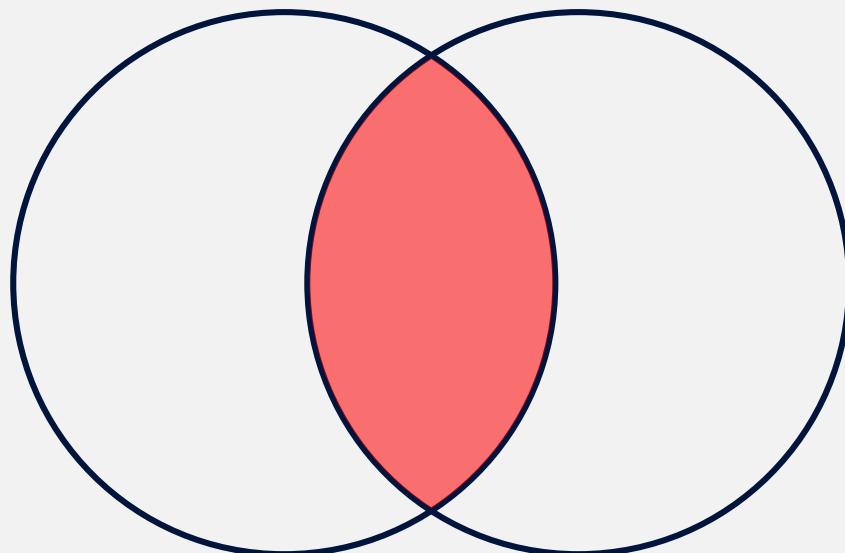
dept_no CHAR(4)

emp_no INT

from_date DATE

to_date DATE

dept_no CHAR(4)



departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)

matching column = linking column

RIGHT JOIN

dept_manager_dup

dept_no CHAR(4)

emp_no INT

from_date DATE

to_date DATE

dept_no CHAR(4)

departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)

matching column = linking column

RIGHT JOIN

- **RIGHT JOIN**

RIGHT JOIN

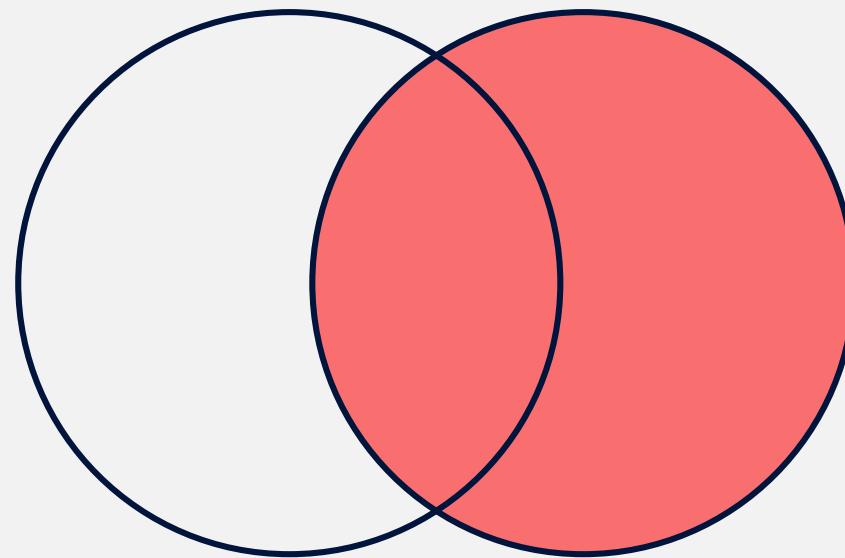
- **RIGHT JOIN**

when applying a **RIGHT JOIN**, all the records from the right table will be included in the result set

RIGHT JOIN

- **RIGHT JOIN**

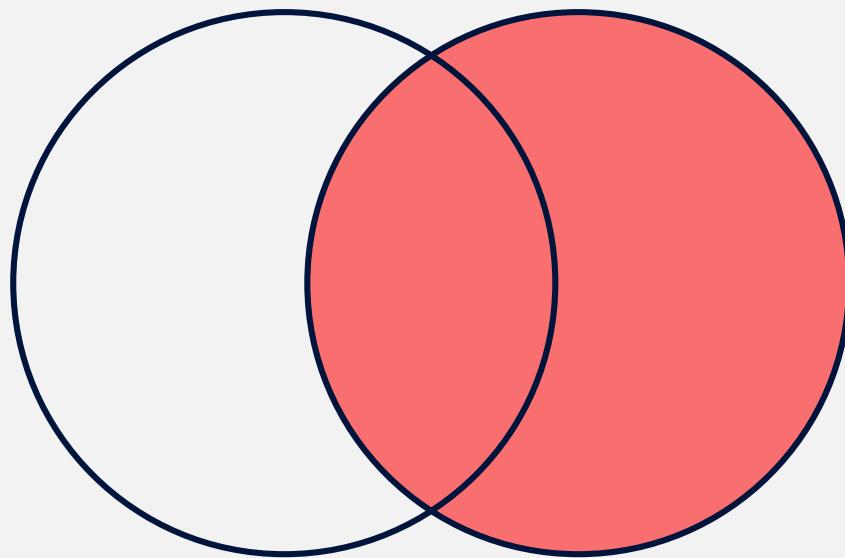
when applying a RIGHT JOIN, all the records from the right table will be included in the result set



RIGHT JOIN

- **RIGHT JOIN**

when applying a **RIGHT JOIN**, all the records from the right table will be included in the result set

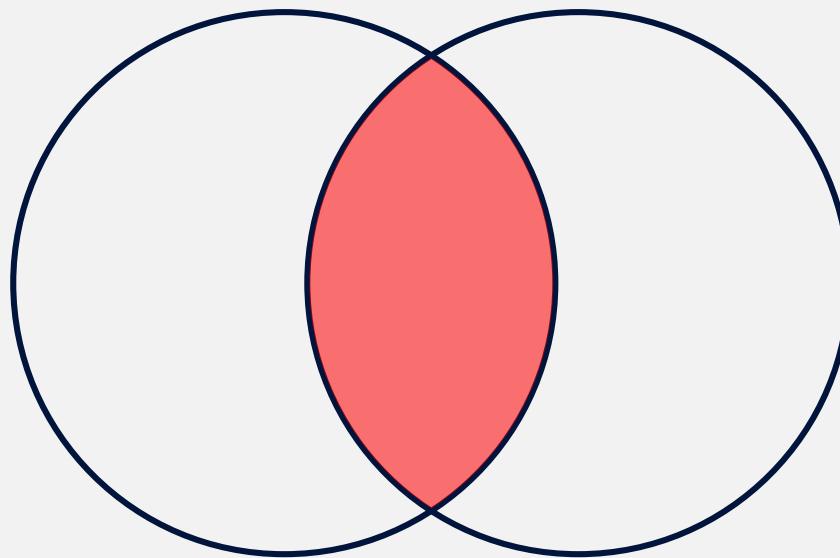


values from the left table will be included only if their *linking column* contains a value coinciding, or *matching*, with a value from the *linking column* of the right table

RIGHT JOIN

- **RIGHT JOIN**

when applying a **RIGHT JOIN**, all the records from the right table will be included in the result set



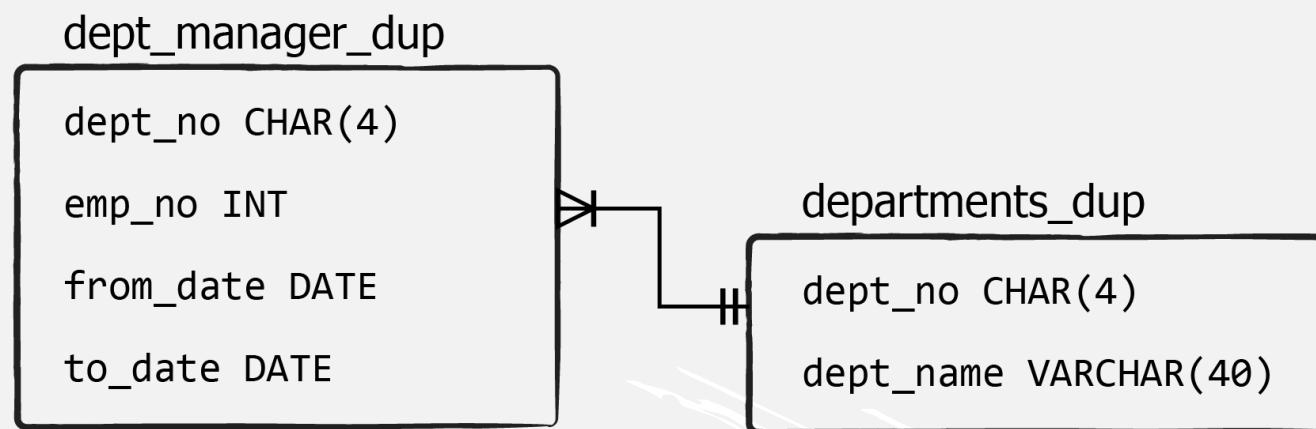
values from the left table will be included only if their *linking column* contains a value coinciding, or *matching*, with a value from the *linking column* of the right table

RIGHT JOIN

- LEFT and RIGHT joins are perfect examples of one-to-many relationships

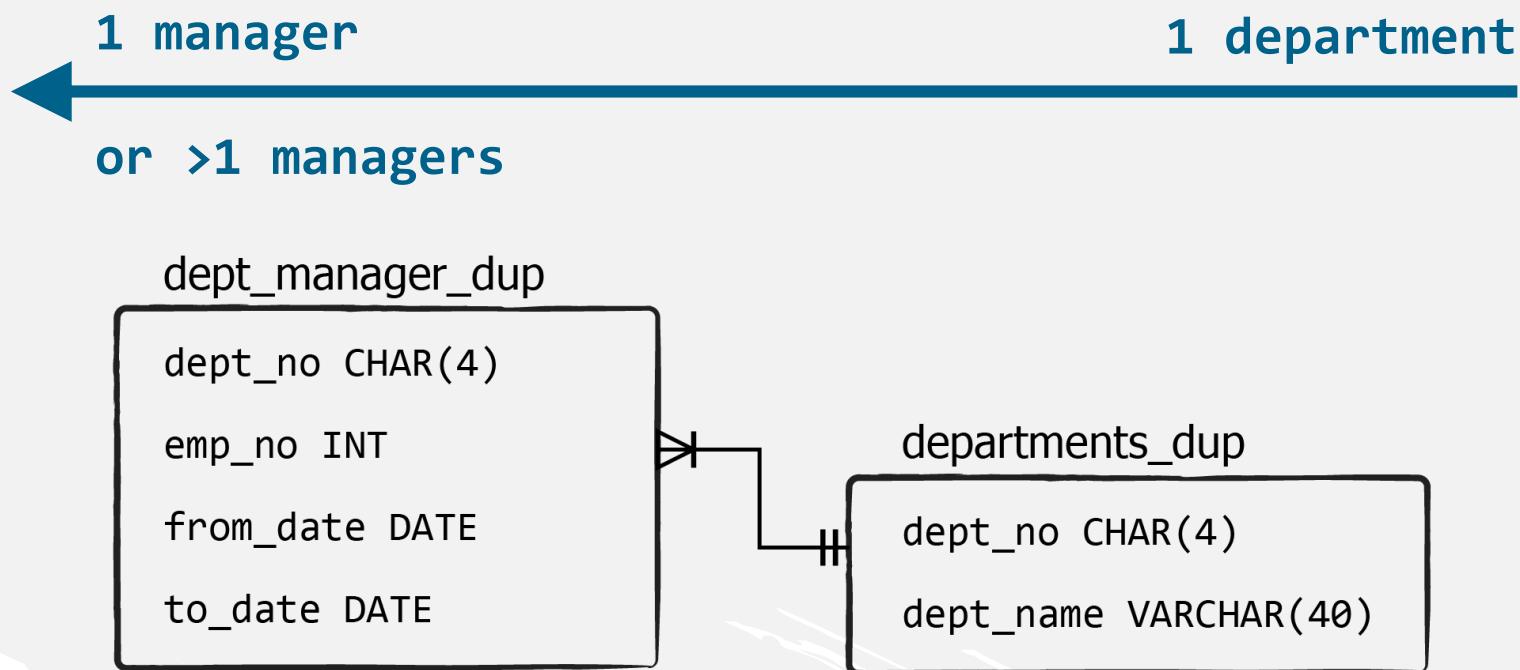
RIGHT JOIN

- LEFT and RIGHT joins are perfect examples of one-to-many relationships



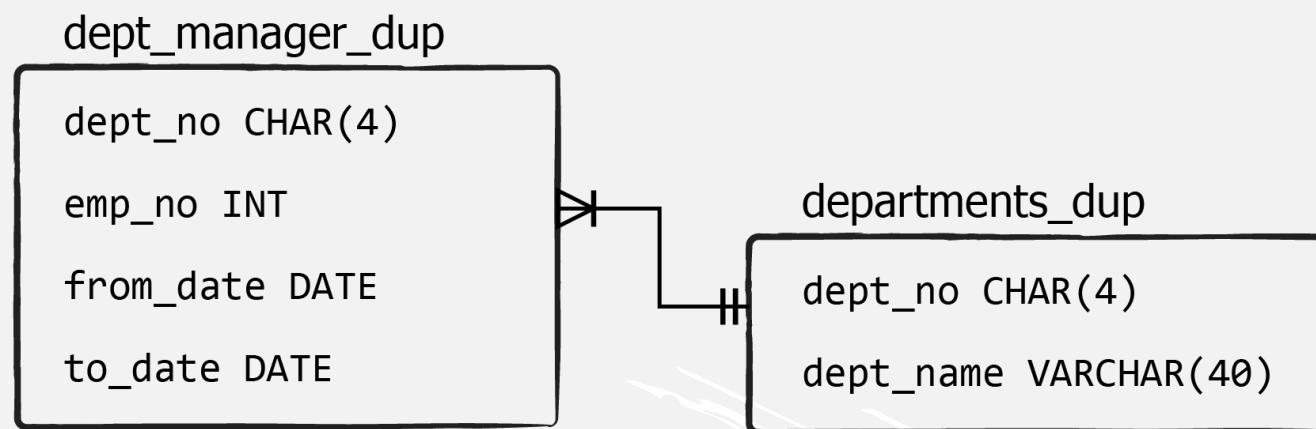
RIGHT JOIN

- LEFT and RIGHT joins are perfect examples of one-to-many relationships



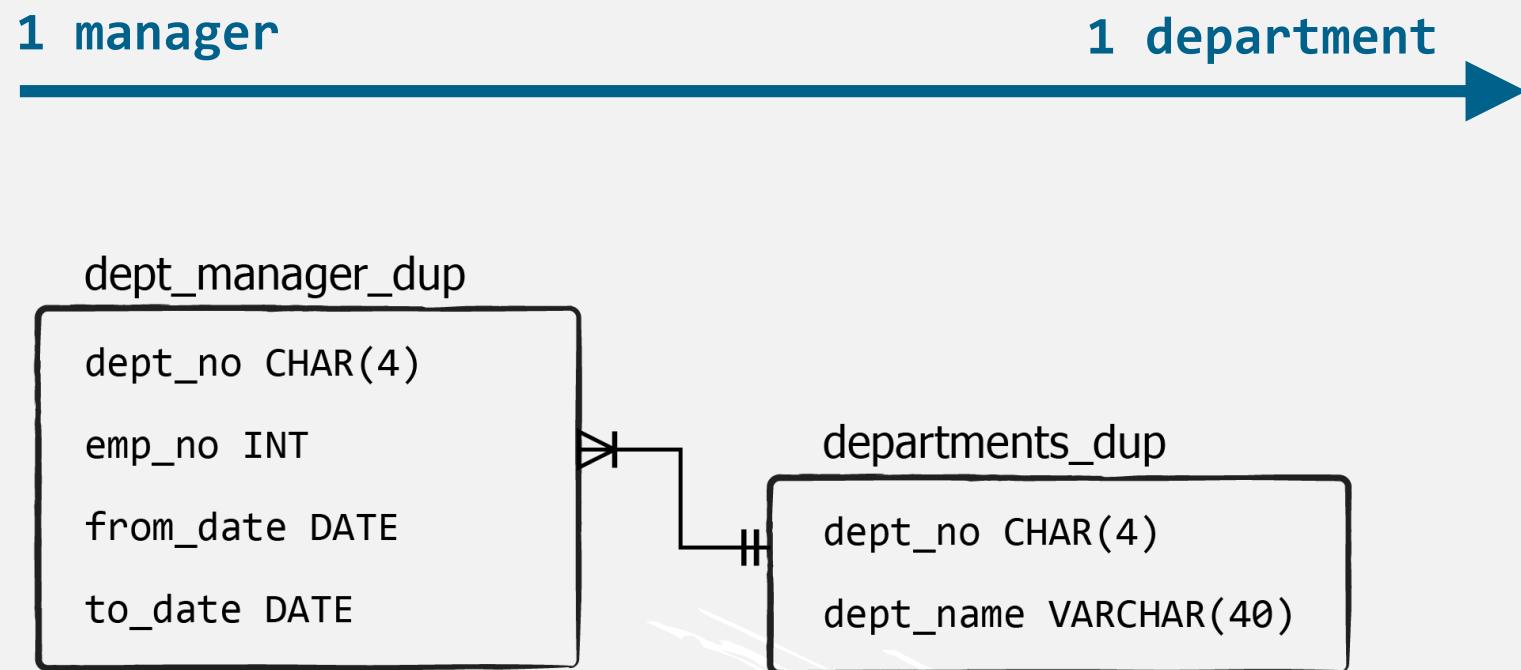
RIGHT JOIN

- LEFT and RIGHT joins are perfect examples of one-to-many relationships



RIGHT JOIN

- LEFT and RIGHT joins are perfect examples of one-to-many relationships



A large, modern conference room is shown from a perspective looking down the length of the room. On the left, there are several rows of light-colored office chairs arranged facing a front wall. The front wall features a large, dark rectangular screen or board. Above the screen, there are two smaller, separate screens. The room has a high ceiling with a grid of recessed lighting. Large windows along the right side provide a view of an outdoor area with some trees and possibly a parking lot. The overall atmosphere is professional and spacious.

The New and the Old Join Syntax

The New and the Old Join Syntax

- Relational Database Essentials
- MySQL Constraints
- SELECT, INSERT, UPDATE, DELETE
- MySQL Aggregate Functions
- INNER JOIN, LEFT JOIN, RIGHT JOIN

The New and the Old Join Syntax

Next:

- Relational Database Essentials
- MySQL Constraints
- SELECT, INSERT, UPDATE, DELETE
- MySQL Aggregate Functions
- INNER JOIN, LEFT JOIN, RIGHT JOIN

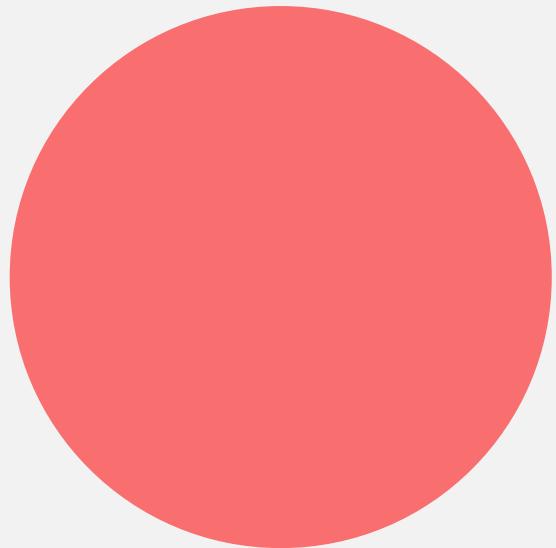
The New and the Old Join Syntax

- Relational Database Essentials
- MySQL Constraints
- SELECT, INSERT, UPDATE, DELETE
- MySQL Aggregate Functions
- INNER JOIN, LEFT JOIN, RIGHT JOIN

Next:

- Subqueries, Self-joins
- Stored Routines
- Advanced SQL Tools

The New and the Old Join Syntax



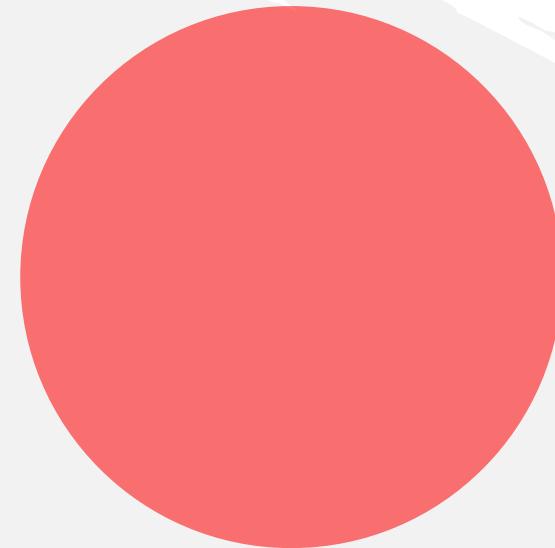
dept_manager_dup

dept_no CHAR(4)

emp_no INT

from_date DATE

to_date DATE



departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)

The New and the Old Join Syntax

dept_manager_dup

dept_no CHAR(4)

emp_no INT

from_date DATE

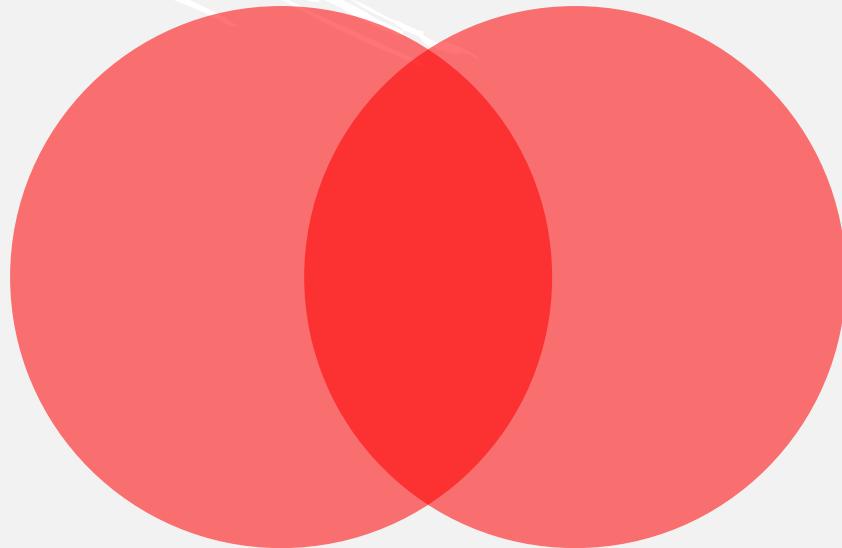
to_date DATE

departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)

The New and the Old Join Syntax



dept_manager_dup

dept_no CHAR(4)

emp_no INT

from_date DATE

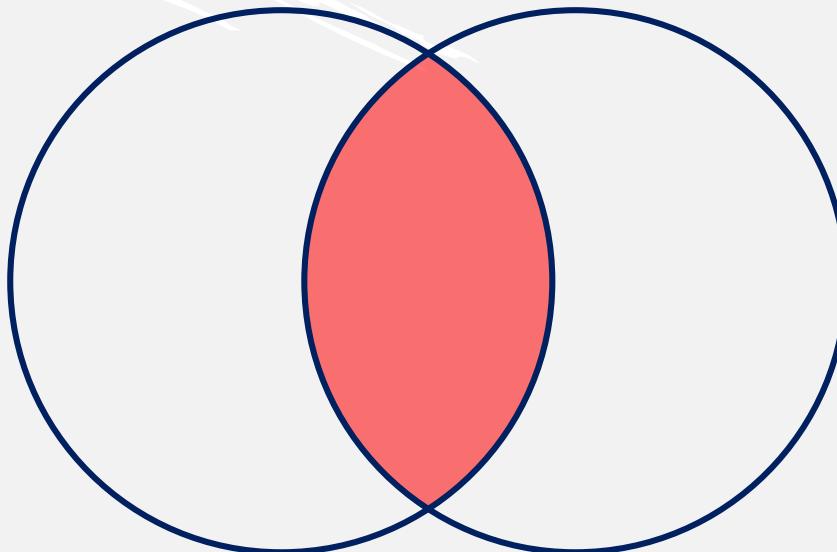
to_date DATE

departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)

The New and the Old Join Syntax



dept_manager_dup

dept_no CHAR(4)

emp_no INT

from_date DATE

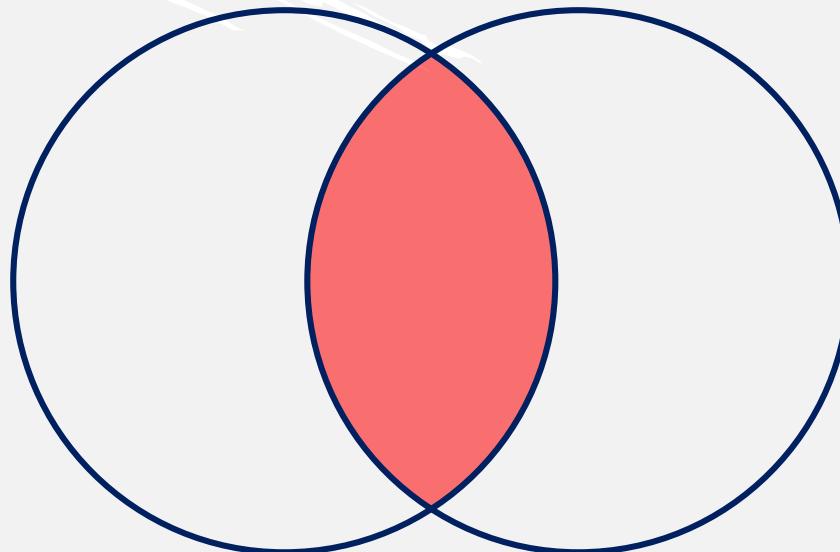
to_date DATE

departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)

The New and the Old Join Syntax



dept_manager_dup

dept_no CHAR(4)

emp_no INT

from_date DATE

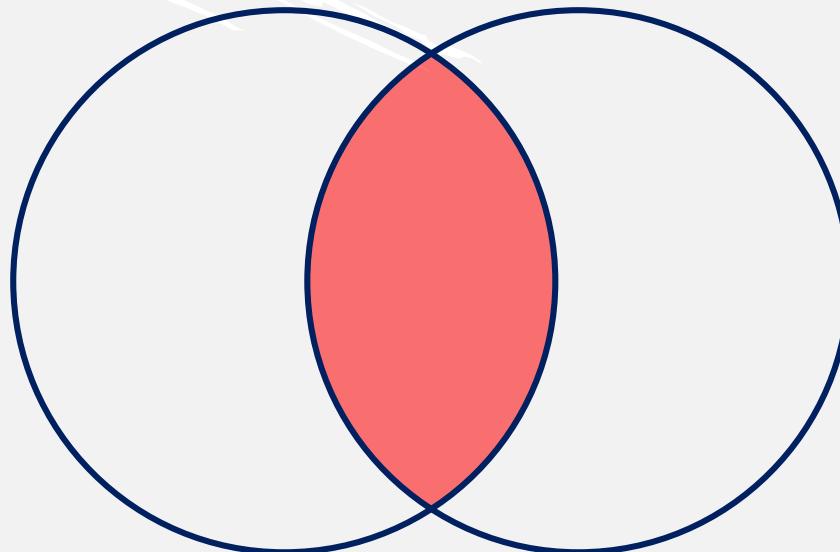
to_date DATE

departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)

The New and the Old Join Syntax



connection points

dept_manager_dup

dept_no CHAR(4)

emp_no INT

from_date DATE

to_date DATE

departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)

The New and the Old Join Syntax

dept_manager_dup

dept_no CHAR(4)

emp_no INT

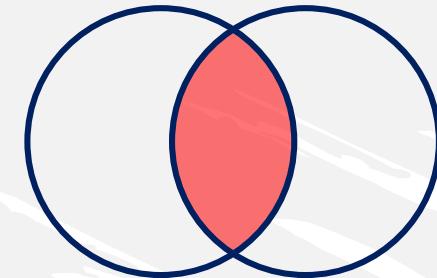
from_date DATE

to_date DATE

departments_dup

dept_no CHAR(4)

dept_name VARCHAR(40)



connection points

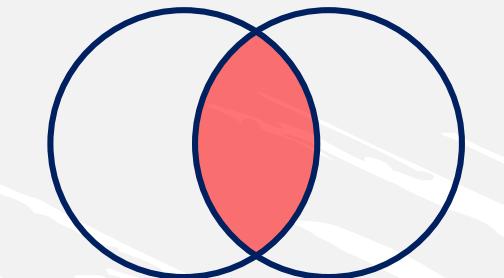
The New and the Old Join Syntax

emp_no	dept_no	from_date	to_date
999904	NULL	2017-01-01	NULL
999905	NULL	2017-01-01	NULL
999906	NULL	2017-01-01	NULL
999907	NULL	2017-01-01	NULL
110085	d002	1985-01-01	1989-12-17
110114	d002	1989-12-17	9999-01-01
110183	d003	1985-01-01	1992-03-21
110228	d003	1992-03-21	9999-01-01
110303	d004	1985-01-01	1988-09-09
110344	d004	1988-09-09	1992-08-02
110386	d004	1992-08-02	1996-08-30
110420	d004	1996-08-30	9999-01-01
110511	d005	1985-01-01	1992-04-25
110567	d005	1992-04-25	9999-01-01
110725	d006	1985-01-01	1989-05-06
110765	d006	1989-05-06	1991-09-12
110800	d006	1991-09-12	1994-06-28
110854	d006	1994-06-28	9999-01-01
111035	d007	1985-01-01	1991-03-07
111133	d007	1991-03-07	9999-01-01
111400	d008	1985-01-01	1991-04-08
111534	d008	1991-04-08	9999-01-01
111692	d009	1985-01-01	1988-10-17
111784	d009	1988-10-17	1992-09-08
111877	d009	1992-09-08	1996-01-03
111939	d009	1996-01-03	9999-01-01

dept_manager_dup

```
dept_no CHAR(4)
emp_no INT
from_date DATE
to_date DATE
```

dept_no	emp_no	dept_name
d003	110228	Human Resources
d003	110183	Human Resources
d004	110344	Production
d004	110420	Production
d004	110303	Production
d004	110386	Production
d005	110567	Development
d005	110511	Development
d006	110800	Quality Management
d006	110765	Quality Management
d006	110854	Quality Management
d006	110725	Quality Management
d007	111035	Sales
d007	111133	Sales
d008	111400	Research
d008	111534	Research
d009	111784	Customer Service
d009	111939	Customer Service
d009	111692	Customer Service
d009	111877	Customer Service



connection points

dept_no	dept_name
NULL	Public Relations
d001	Marketing
d003	Human Resources
d004	Production
d005	Development
d006	Quality Management
d007	Sales
d008	Research
d009	Customer Service
d010	NULL
d011	NULL

departments_dup

```
dept_no CHAR(4)
dept_name VARCHAR(40)
```

The New and the Old Join Syntax

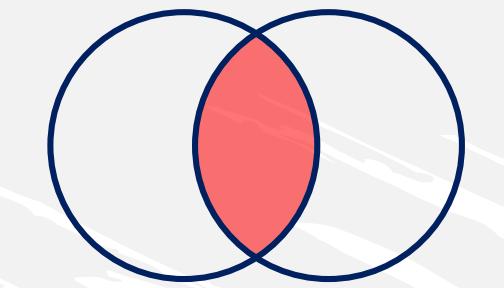
emp_no	dept_no	from_date	to_date
999904	NULL	X	2017-01-01
999905	NULL	X	2017-01-01
999906	NULL	X	2017-01-01
999907	NULL	X	2017-01-01
110085	d002	X	1985-01-01
110114	d002	X	1989-12-17
110183	d003	✓	1985-01-01
110228	d003	✓	1992-03-21
110303	d004	✓	1985-01-01
110344	d004	✓	1988-09-09
110386	d004	✓	1992-08-02
110420	d004	✓	1996-08-30
110511	d005	✓	1985-01-01
110567	d005	✓	1992-04-25
110725	d006	✓	1985-01-01
110765	d006	✓	1989-05-06
110800	d006	✓	1991-09-12
110854	d006	✓	1994-06-28
111035	d007	✓	1985-01-01
111133	d007	✓	1991-03-07
111400	d008	✓	1985-01-01
111534	d008	✓	1991-04-08
111692	d009	✓	1985-01-01
111784	d009	✓	1988-10-17
111877	d009	✓	1992-09-08
111939	d009	✓	1996-01-03

dept_manager_dup

```
dept_no CHAR(4)
emp_no INT
from_date DATE
to_date DATE
```

dept_no	emp_no	dept_name
d003	110228	Human Resources
d003	110183	Human Resources
d004	110344	Production
d004	110420	Production
d004	110303	Production
d004	110386	Production
d005	110567	Development
d005	110511	Development
d006	110800	Quality Management
d006	110765	Quality Management
d006	110854	Quality Management
d006	110725	Quality Management
d007	111035	Sales
d007	111133	Sales
d008	111400	Research
d008	111534	Research
d009	111784	Customer Service
d009	111939	Customer Service
d009	111692	Customer Service
d009	111877	Customer Service

dept_no	dept_name
NULL	Public Relations
d001	Marketing
d003	Human Resources
d004	Production
d005	Development
d006	Quality Management
d007	Sales
d008	Research
d009	Customer Service
d010	NULL
d011	NULL



connection points

departments_dup

```
dept_no CHAR(4)
dept_name VARCHAR(40)
```

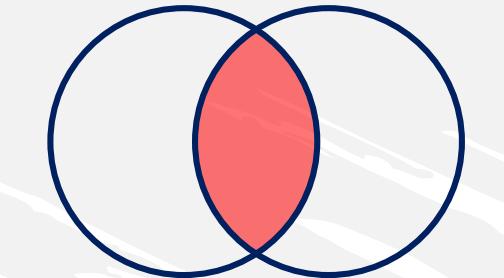
The New and the Old Join Syntax

emp_no	dept_no	from_date	to_date
999904	NULL	2017-01-01	NULL
999905	NULL	2017-01-01	NULL
999906	NULL	2017-01-01	NULL
999907	NULL	2017-01-01	NULL
110085	d002	1985-01-01	1989-12-17
110114	d002	1989-12-17	9999-01-01
110183	d003	1985-01-01	1992-03-21
110228	d003	1992-03-21	9999-01-01
110303	d004	1985-01-01	1988-09-09
110344	d004	1988-09-09	1992-08-02
110386	d004	1992-08-02	1996-08-30
110420	d004	1996-08-30	9999-01-01
110511	d005	1985-01-01	1992-04-25
110567	d005	1992-04-25	9999-01-01
110725	d006	1985-01-01	1989-05-06
110765	d006	1989-05-06	1991-09-12
110800	d006	1991-09-12	1994-06-28
110854	d006	1994-06-28	9999-01-01
111035	d007	1985-01-01	1991-03-07
111133	d007	1991-03-07	9999-01-01
111400	d008	1985-01-01	1991-04-08
111534	d008	1991-04-08	9999-01-01
111692	d009	1985-01-01	1988-10-17
111784	d009	1988-10-17	1992-09-08
111877	d009	1992-09-08	1996-01-03
111939	d009	1996-01-03	9999-01-01

dept_manager_dup

```
dept_no CHAR(4)
emp_no INT
from_date DATE
to_date DATE
```

dept_no	emp_no	dept_name
d003	110228	Human Resources
d003	110183	Human Resources
d004	110344	Production
d004	110420	Production
d004	110303	Production
d004	110386	Production
d005	110567	Development
d005	110511	Development
d006	110800	Quality Management
d006	110765	Quality Management
d006	110854	Quality Management
d006	110725	Quality Management
d007	111035	Sales
d007	111133	Sales
d008	111400	Research
d008	111534	Research
d009	111784	Customer Service
d009	111939	Customer Service
d009	111692	Customer Service
d009	111877	Customer Service



connection points

dept_no	dept_name
NULL	Public Relations
d001	Marketing
d003	Human Resources
d004	Production
d005	Development
d006	Quality Management
d007	Sales
d008	Research
d009	Customer Service
d010	NULL
d011	NULL

departments_dup

```
dept_no CHAR(4)
dept_name VARCHAR(40)
```

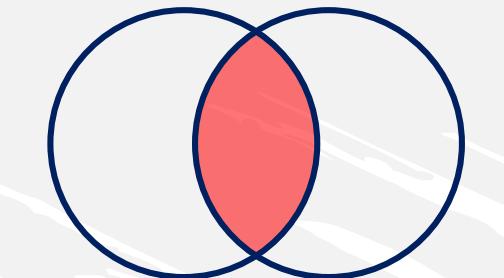
The New and the Old Join Syntax

emp_no	dept_no	from_date	to_date
999904	NULL	2017-01-01	NULL
999905	NULL	2017-01-01	NULL
999906	NULL	2017-01-01	NULL
999907	NULL	2017-01-01	NULL
110085	d002	1985-01-01	1989-12-17
110114	d002	1989-12-17	9999-01-01
110183	d003	1985-01-01	1992-03-21
110228	d003	1992-03-21	9999-01-01
110303	d004	1985-01-01	1988-09-09
110344	d004	1988-09-09	1992-08-02
110386	d004	1992-08-02	1996-08-30
110420	d004	1996-08-30	9999-01-01
110511	d005	1985-01-01	1992-04-25
110567	d005	1992-04-25	9999-01-01
110725	d006	1985-01-01	1989-05-06
110765	d006	1989-05-06	1991-09-12
110800	d006	1991-09-12	1994-06-28
110854	d006	1994-06-28	9999-01-01
111035	d007	1985-01-01	1991-03-07
111133	d007	1991-03-07	9999-01-01
111400	d008	1985-01-01	1991-04-08
111534	d008	1991-04-08	9999-01-01
111692	d009	1985-01-01	1988-10-17
111784	d009	1988-10-17	1992-09-08
111877	d009	1992-09-08	1996-01-03
111939	d009	1996-01-03	9999-01-01

dept_manager_dup

```
dept_no CHAR(4)
emp_no INT
from_date DATE
to_date DATE
```

dept_no	emp_no	dept_name
d003	110228	Human Resources
d003	110183	Human Resources
d004	110344	Production
d004	110420	Production
d004	110303	Production
d004	110386	Production
d005	110567	Development
d005	110511	Development
d006	110800	Quality Management
d006	110765	Quality Management
d006	110854	Quality Management
d006	110725	Quality Management
d007	111035	Sales
d007	111133	Sales
d008	111400	Research
d008	111534	Research
d009	111784	Customer Service
d009	111939	Customer Service
d009	111692	Customer Service
d009	111877	Customer Service



connection points

dept_no	dept_name
NULL	Public Relations
d001	Marketing
d003	Human Resources
d004	Production
d005	Development
d006	Quality Management
d007	Sales
d008	Research
d009	Customer Service
d010	NULL
d011	NULL

departments_dup

```
dept_no CHAR(4)
dept_name VARCHAR(40)
```

The New and the Old Join Syntax

- WHERE (*the Old Join Syntax*)



SQL

```
SELECT
    t1.column_name, t1.column_name, ..., t2.column_name, ...
FROM
    table_1 t1,
    table_2 t2
WHERE
    t1.column_name = t2.column_name;
```

The New and the Old Join Syntax

- JOIN or WHERE?

The New and the Old Join Syntax

- **JOIN or WHERE?**

- the retrieved output is *identical*

The New and the Old Join Syntax

- **JOIN or WHERE?**

- the retrieved output is *identical*
- using WHERE is more *time-consuming*

The New and the Old Join Syntax

- **JOIN or WHERE?**

- the retrieved output is *identical*
- using WHERE is more *time-consuming*
- the WHERE syntax is perceived as *morally old* and is rarely employed by professionals

The New and the Old Join Syntax

- **JOIN or WHERE?**

- the retrieved output is *identical*
- using WHERE is more *time-consuming*
- the WHERE syntax is perceived as *morally old* and is rarely employed by professionals
- the JOIN syntax allows you to modify the connection between tables easily

The New and the Old Join Syntax

JOIN (*the New Join Syntax*) vs WHERE (*the Old Join Syntax*)



JOIN and WHERE Used Together

JOIN and WHERE Used Together

- JOIN + WHERE

JOIN and WHERE Used Together

- **JOIN + WHERE**

JOIN:

- used for connecting the “employees” and “salaries” tables

JOIN and WHERE Used Together

- **JOIN + WHERE**

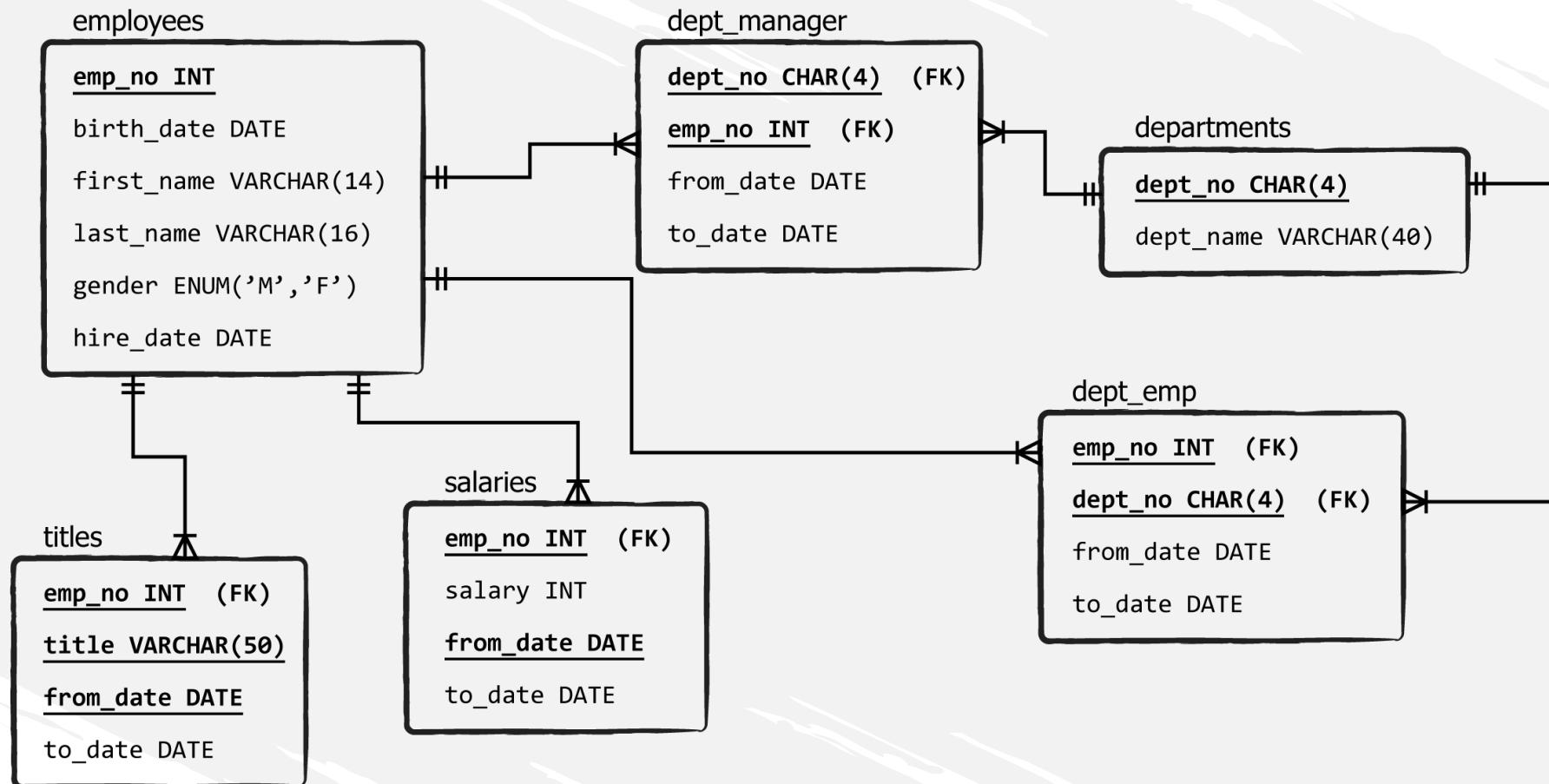
JOIN:

- used for connecting the “employees” and “salaries” tables

WHERE:

- used to define the condition or conditions that will determine which will be the connecting points between the two tables

JOIN and WHERE Used Together





The Functionality of CROSS JOIN

CROSS JOIN

CROSS JOIN

- a cross join will take the values from a certain table and connect them with all the values from the tables we want to join it with

CROSS JOIN

- a cross join will take the values from a certain table and connect them with all the values from the tables we want to join it with
- INNER JOIN

CROSS JOIN

- a cross join will take the values from a certain table and connect them with all the values from the tables we want to join it with
- INNER JOIN
 - typically connects *only the matching values*

CROSS JOIN

- a cross join will take the values from a certain table and connect them with all the values from the tables we want to join it with
- INNER JOIN
 - typically connects *only the matching values*
- CROSS JOIN

CROSS JOIN

- a cross join will take the values from a certain table and connect them with all the values from the tables we want to join it with
- INNER JOIN
 - typically connects *only the matching values*
- CROSS JOIN
 - connects *all the values*, not just those that match

CROSS JOIN

- a cross join will take the values from a certain table and connect them with all the values from the tables we want to join it with

- INNER JOIN

- typically connects *only the matching values*

- CROSS JOIN

- connects *all the values*, not just those that match
 - the Cartesian product of the values of two or more sets

CROSS JOIN

- a cross join will take the values from a certain table and connect them with all the values from the tables we want to join it with
- INNER JOIN
 - typically connects *only the matching values*
- CROSS JOIN
 - connects *all the values*, not just those that match
 - the Cartesian product of the values of two or more sets
 - particularly useful when the tables in a database are not well connected

FRAGILE

HANDLE WITH CARE



JOIN More Than Two Tables in SQL

JOIN More Than Two Tables in SQL

- when creating a query that joins multiple tables, you must back it with *strong intuition* and a *crystal-clear idea* of how you would like the tables to be connected

JOIN More Than Two Tables in SQL

first_name	last_name

JOIN More Than Two Tables in SQL

first_name	last_name	hire_date

JOIN More Than Two Tables in SQL

first_name	last_name	hire_date	from_date

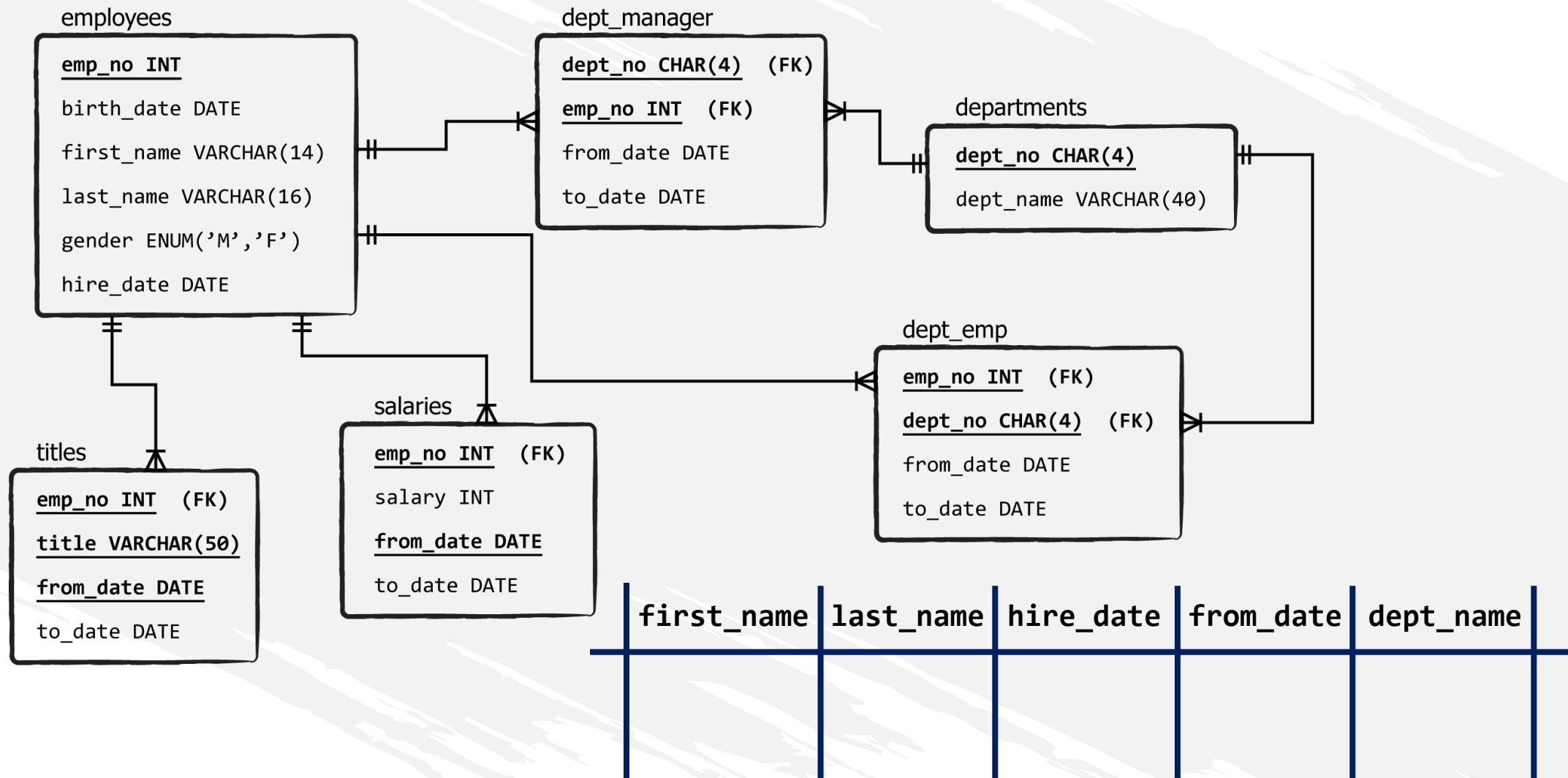
JOIN More Than Two Tables in SQL

first_name	last_name	hire_date	from_date	dept_name

JOIN More Than Two Tables in SQL

first_name	last_name	hire_date	from_date	dept_name

JOIN More Than Two Tables in SQL



A large, modern conference room is shown from a perspective looking down the length of the room. On both sides, there are rows of black office chairs facing towards the front. The front wall features several large, dark rectangular panels, likely projection screens or monitors. The room has a high ceiling with a grid of recessed lighting fixtures. The floor is a light-colored wood or laminate. The overall atmosphere is professional and spacious.

Tips and Tricks for Joins

Tips and Tricks for Joins

dept_name

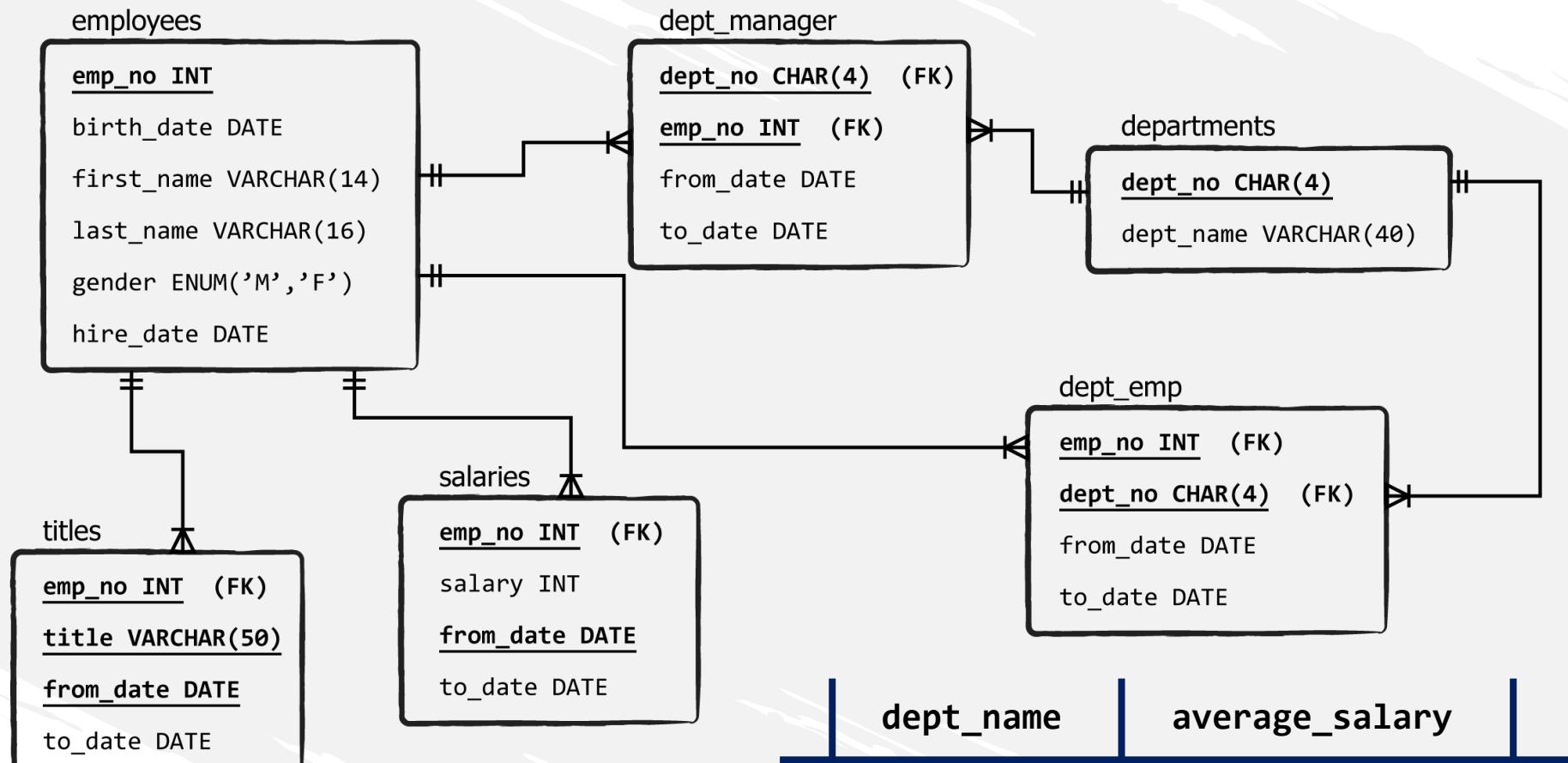
Tips and Tricks for Joins

dept_name	average_salary

Tips and Tricks for Joins

dept_name	average_salary

Tips and Tricks for Joins



dept_name	average_salary

Tips and Tricks for Joins

- JOINS

Tips and Tricks for Joins

JOINS

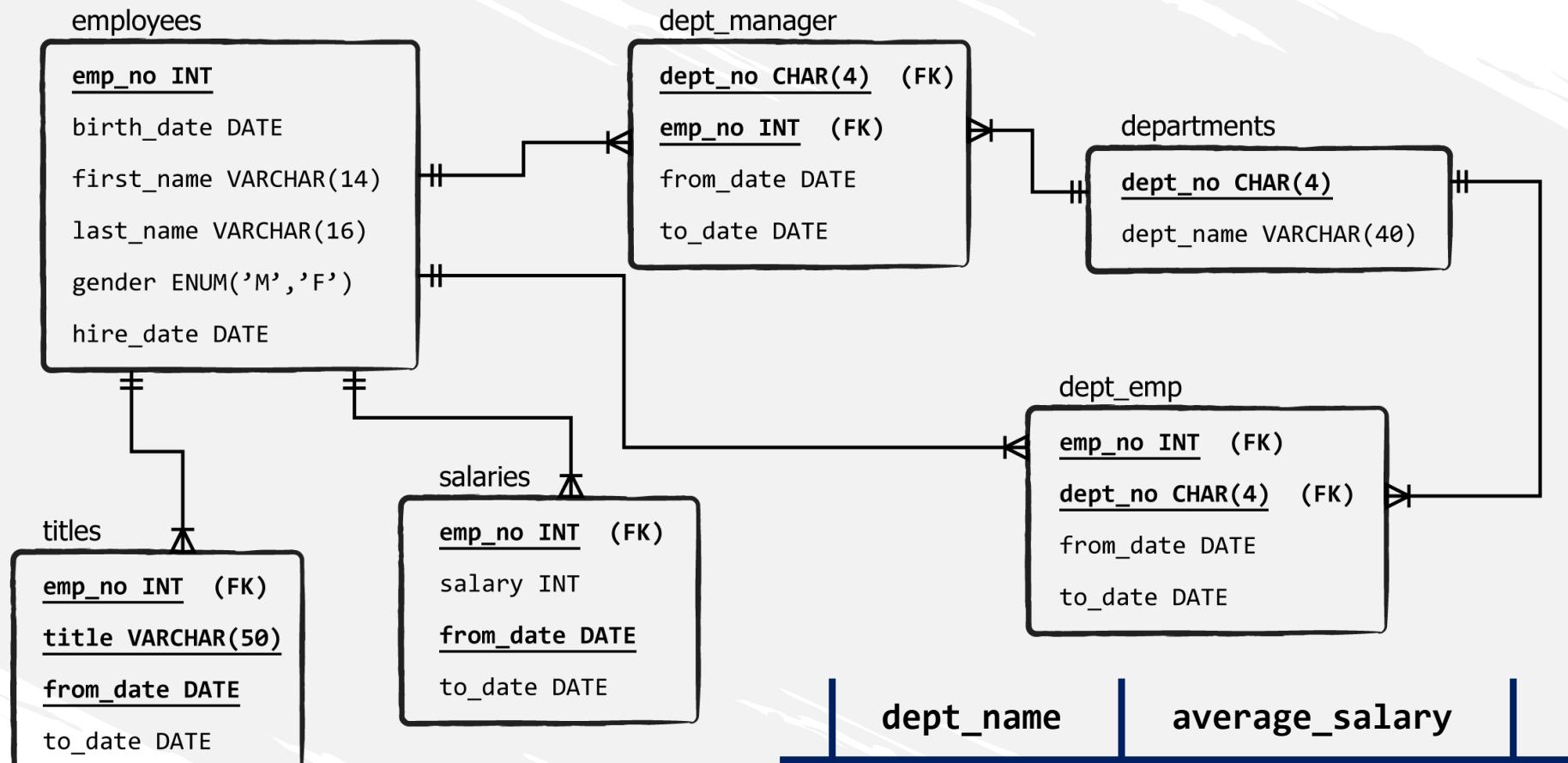
- one should look for key columns, which are common between *the tables involved in the analysis* and are necessary to solve the task at hand

Tips and Tricks for Joins

JOINS

- one should look for key columns, which are common between *the tables involved in the analysis* and are necessary to solve the task at hand
- these columns do not need to be foreign or private keys

Tips and Tricks for Joins



dept_name	average_salary

A large conference room with rows of chairs facing a central table. The room has a modern design with a grid-patterned ceiling and large windows overlooking a landscape.

UNION vs UNION ALL

UNION vs UNION ALL

- UNION ALL

UNION vs UNION ALL

- **UNION ALL**

used to combine a few SELECT statements in a single output

UNION vs UNION ALL

- **UNION ALL**

used to combine a few SELECT statements in a single output

- you can think of it as a tool that allows you to *unify tables*

UNION vs UNION ALL

- **UNION ALL**

used to combine a few SELECT statements in a single output

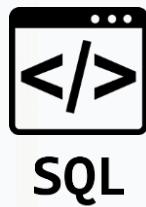


```
SELECT  
      N columns  
FROM  
      table_1  
UNION ALL SELECT  
      N columns  
FROM  
      table_2;
```

UNION vs UNION ALL

- **UNION ALL**

used to combine a few SELECT statements in a single output



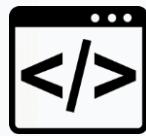
```
SELECT  
      N columns  
FROM  
      table_1  
UNION ALL SELECT  
      N columns  
FROM  
      table_2;
```

We have to select the same number of columns from each table.

UNION vs UNION ALL

• UNION ALL

used to combine a few SELECT statements in a single output



```
SELECT  
      N columns  
FROM  
      table_1  
UNION ALL SELECT  
      N columns  
FROM  
      table_2;
```

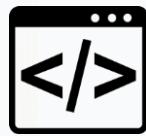
We have to select the same number of columns from each table.

These columns should have the same name,

UNION vs UNION ALL

• UNION ALL

used to combine a few SELECT statements in a single output



```
SELECT  
      N columns  
FROM  
      table_1  
UNION ALL SELECT  
      N columns  
FROM  
      table_2;
```

We have to select the same number of columns from each table.

*These columns should have the same name,
should be in the same order,*

UNION vs UNION ALL

• UNION ALL

used to combine a few SELECT statements in a single output



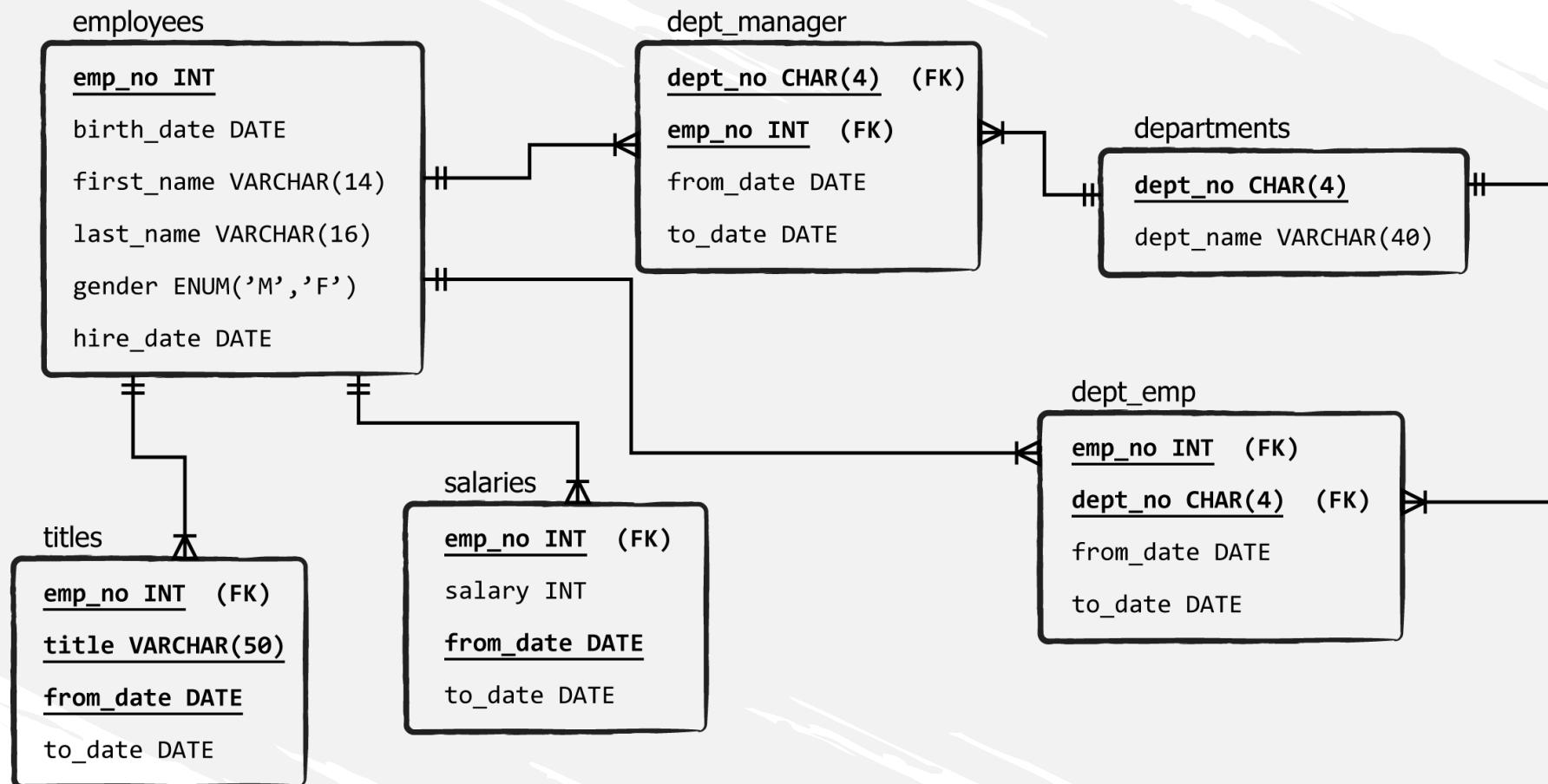
SQL

```
SELECT  
      N columns  
FROM  
      table_1  
UNION ALL SELECT  
      N columns  
FROM  
      table_2;
```

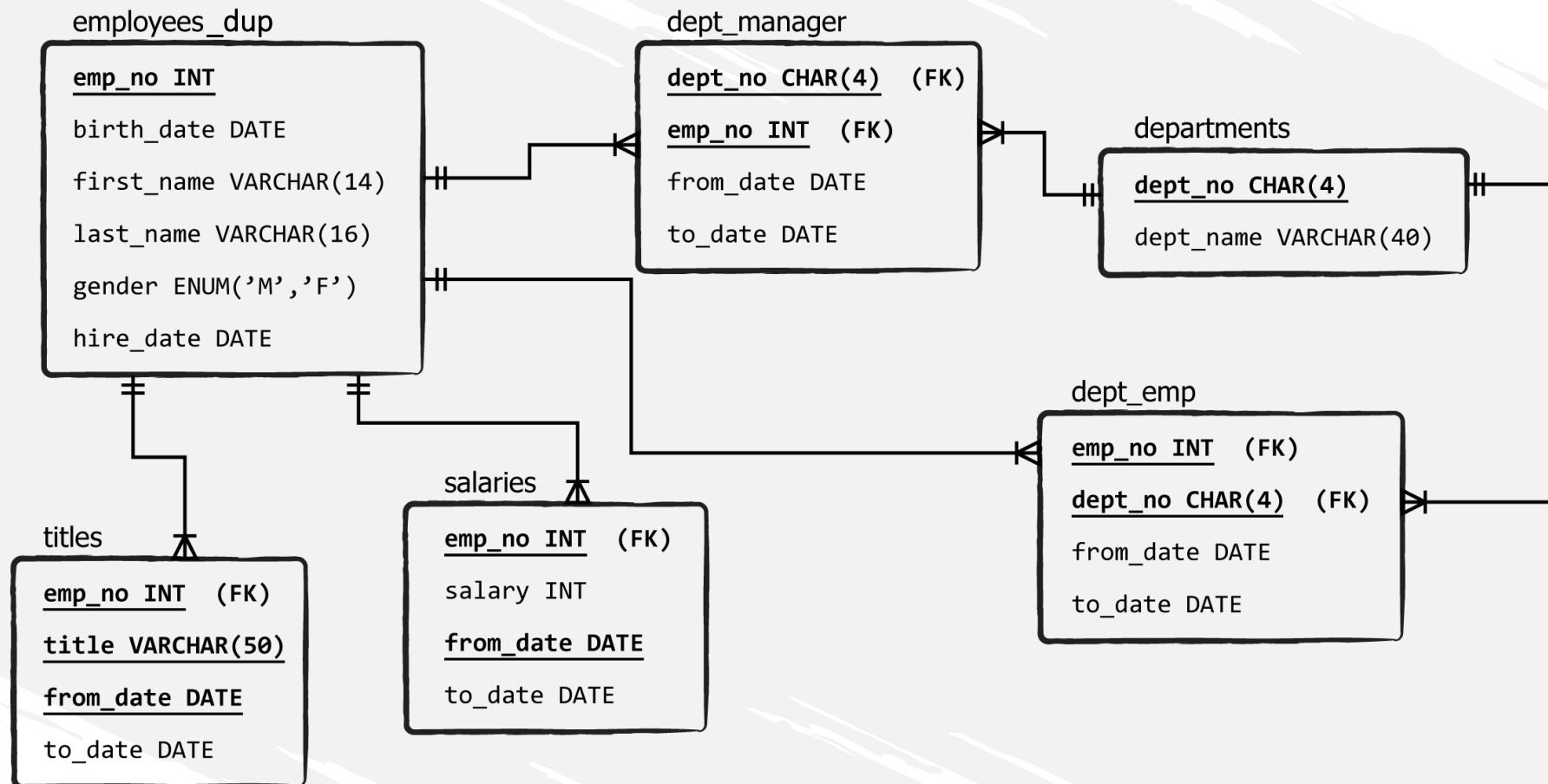
We have to select the same number of columns from each table.

These columns should have the same name, should be in the same order, and should contain related data types.

UNION VS UNION ALL



UNION VS UNION ALL



UNION vs UNION ALL

• UNION



```
SELECT  
      N columns  
FROM  
      table_1  
UNION SELECT  
      N columns  
FROM  
      table_2;
```

UNION vs UNION ALL

- when uniting two identically organized tables

UNION vs UNION ALL

- when uniting two identically organized tables
 - UNION displays only distinct values in the output

UNION vs UNION ALL

- when uniting two identically organized tables
 - UNION displays only distinct values in the output
 - UNION ALL retrieves the duplicates as well

UNION vs UNION ALL

- when uniting two identically organized tables

- UNION displays only distinct values in the output
- UNION ALL retrieves the duplicates as well

UNION vs UNION ALL

- when uniting two identically organized tables

- UNION displays only distinct values in the output
 - UNION uses more MySQL resources
- UNION ALL retrieves the duplicates as well

UNION vs UNION ALL

- when uniting two identically organized tables

- UNION displays only distinct values in the output
 - UNION uses more MySQL resources (computational power and storage space)
- UNION ALL retrieves the duplicates as well

UNION vs UNION ALL

- Looking for better results?

UNION vs UNION ALL

- Looking for better results?

- use UNION

UNION vs UNION ALL

- Looking for better results?
 - use UNION
- Seeking to optimize performance?

UNION vs UNION ALL

- Looking for better results?
 - use UNION
- Seeking to optimize performance?
 - opt for UNION ALL