WELCOME

CSCA20 Week 11

Assignment 2

DUE TONIGHT!

Don’t submit last minute!

Introduction to Databases 2

Databases Basics

Review

What is a Database?

A database is just a well-structured collection of data.

Data should be easily stored and retrieved

Often data is stored in the form of tables where the headers are properties, and each row represents an entry

Writing A Query

A query defines the parameters for the search that we want to perform on a database.

```
SELECT [some attribute or column]
FROM   [some table]
WHERE  [some condition is true]
```

Depending on the version of SQL that you use, the exact syntax will vary, but the idea is always the same.

Working With A Database

The first thing we need to do is **import** the sqlite3 module.

```
import sqlite3
```

Next, we need to **connect** to our database
and **link** to it using a cursor. Now we can do some **work**.

```
connection = sqlite3.connect(name of database)
cursor = connection.cursor()
```

Once we are done making changes, we need to **save**.

```
connection.commit()
```

After all changes have been saved, **close** all connections.

```
cursor.close()
connection.close()
```

Manipulating the Database

The cursor is a link to your database.
in other words, if you want to do something to your database, you must reference it using the cursor.

If you want to think of the database as a Object like a String, List, Dictionary etc, then the cursor is the database object that contains a set of database tools.

.execute()

The database cursor’s execute method isn’t a conventional method like those that you’re used to seeing. It doesn’t do any one thing…

.execute() does to the database whatever you tell it to do in SQL!

In other words, it’s the bridge between your Python code and the SQL that modifies the database.

`.execute()`

SQL queries are always written and passed to `.execute()` as a string.

The SQL itself specifies the operation that `.execute()` performs on your database!

Common Table Tasks

Here are some common tasks that can be done using SQL and the cursor's `.execute()` method:

```
DROP TABLE IF EXISTS table_name
```

If the table already exists, erase it and set it up all over again. This should be used inside your functions before you create any new table.

```
CREATE TABLE table_name(columnName TYPE …)
```

Creates a new table with the given name and columns.

Columns must indicate the names of each column and the type of data that should go into that column. These types are not the same across Python and SQL!

Python Type	SQL Type
`Str`	`TEXT`
`Float`	`REAL`
`Int`	`INTEGER`

Common Table Tasks

```
SELECT columns FROM table WHERE condition
```

SQL queries (Of the format we discussed earlier) Can also be passed into .execute(), indicating that we want to search the database.

```
(INSERT INTO table VALUES (?, ?, …), data)
```

Add an entry (also called a VALUE or row) into the table. This is the query that must be paired with an actual dataset. Each “?” Is a placeholder for an attribute of the actual dataset.

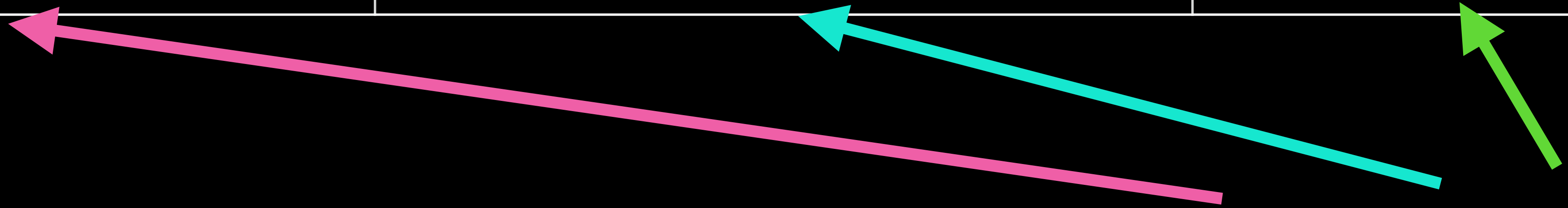
```
(INSERT INTO table VALUES (?, ?, …), data)
```

Suppose we have a table called Uploads:

image_name	uploader	image_size
“img_1134.png”	“mrBubbles123”	30
“img_6126.jpg”	“hanna_mclean”	13

We want to add this row:

“dsc_2342.tiff”	“ms_skittles”	45



```
query = “INSERT INTO uploads VALUES (?, ?, ?)”
data = (“dsc_2342.tiff”, “ms_skittles”, 45)
cursor.execute(query, data)
```

table joins

When Are Joins Useful?

Sometimes the information we need is Spread across more than one table

We need some way to relate this data in a way that makes sense, and is still easy to access

Solution:

We can combine multiple smaller tables into a single larger table that contains all the information we want!

Types of Joins

There are many types of joins that we can use depending on the data we're working with:

In lecture, David talked about:

- Left Joins
- Right Joins (Not In SQLite)
- Inner Joins
- Full Outer Joins (Not In SQLite)
- Cross Joins (This one is very different!)

Types of Joins

There are many types of joins that we can use depending on the data we're working with:

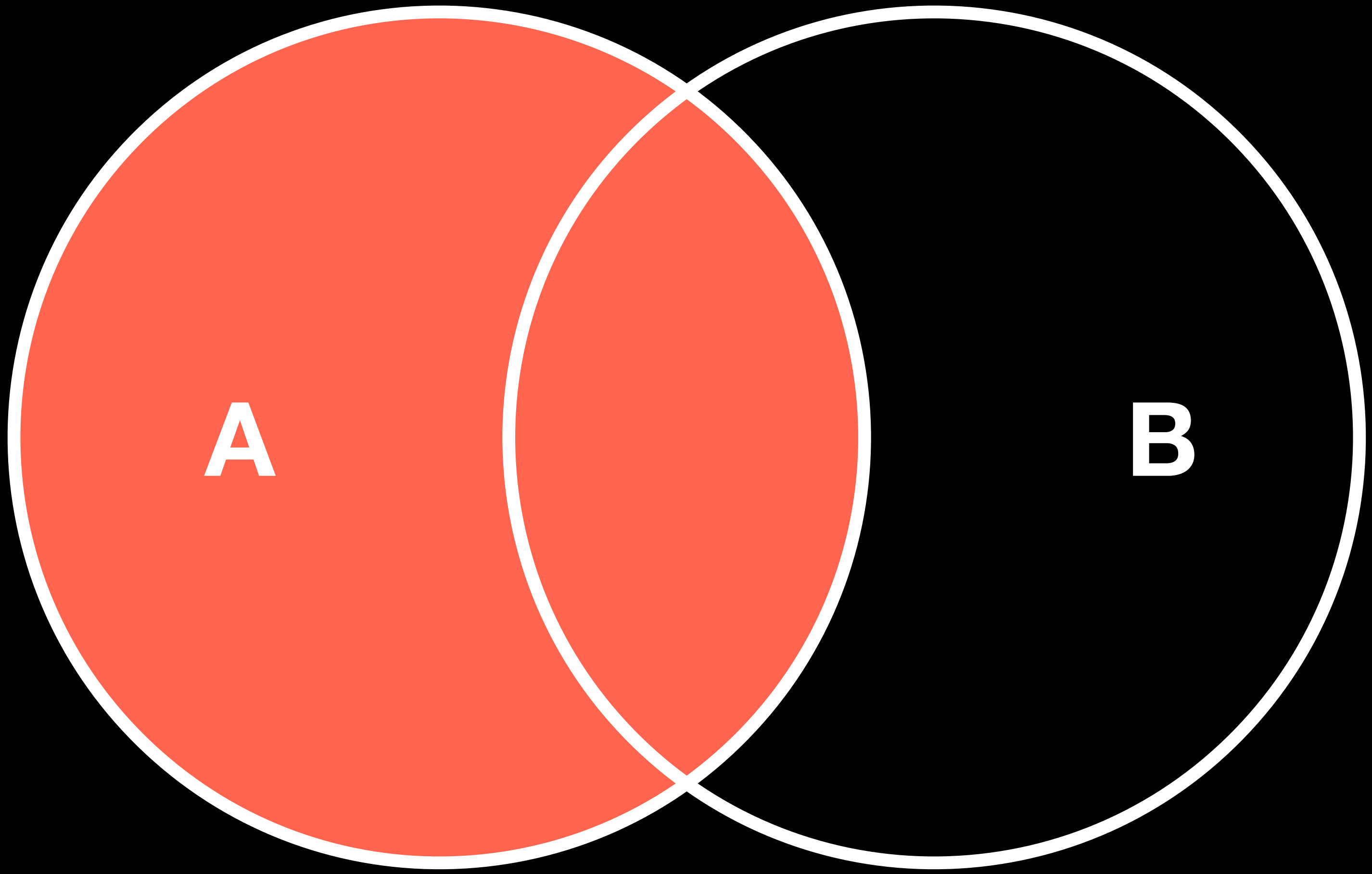
In lecture, David talked about:

- Left Joins
- Right Joins (Not In SQLite)
- Inner Joins
- Full Outer Joins (Not In SQLite)

We'll talk about these today.

Cross Joins

Left Join



```
SELECT [some attribute or column]
FROM   A LEFT JOIN B
ON     A.key = B.key
WHERE  [some condition is true]
```

Left Join

A =

Movie	Year
Titanic	1997
Avatar	2009

B =

Movie	Genre
Avatar	Action
Grown Ups	Comedy

```
SELECT ...
FROM   A LEFT JOIN B
ON     A.movie = B.movie
WHERE  ...
```

Left Join

A =

Movie	Year
Titanic	1997
Avatar	2009

B =

Movie	Genre
Avatar	Action
Grown Ups	Comedy

```
SELECT …
FROM    A LEFT JOIN B
ON      A.movie = B.movie
WHERE   …
```

Left Join

A =

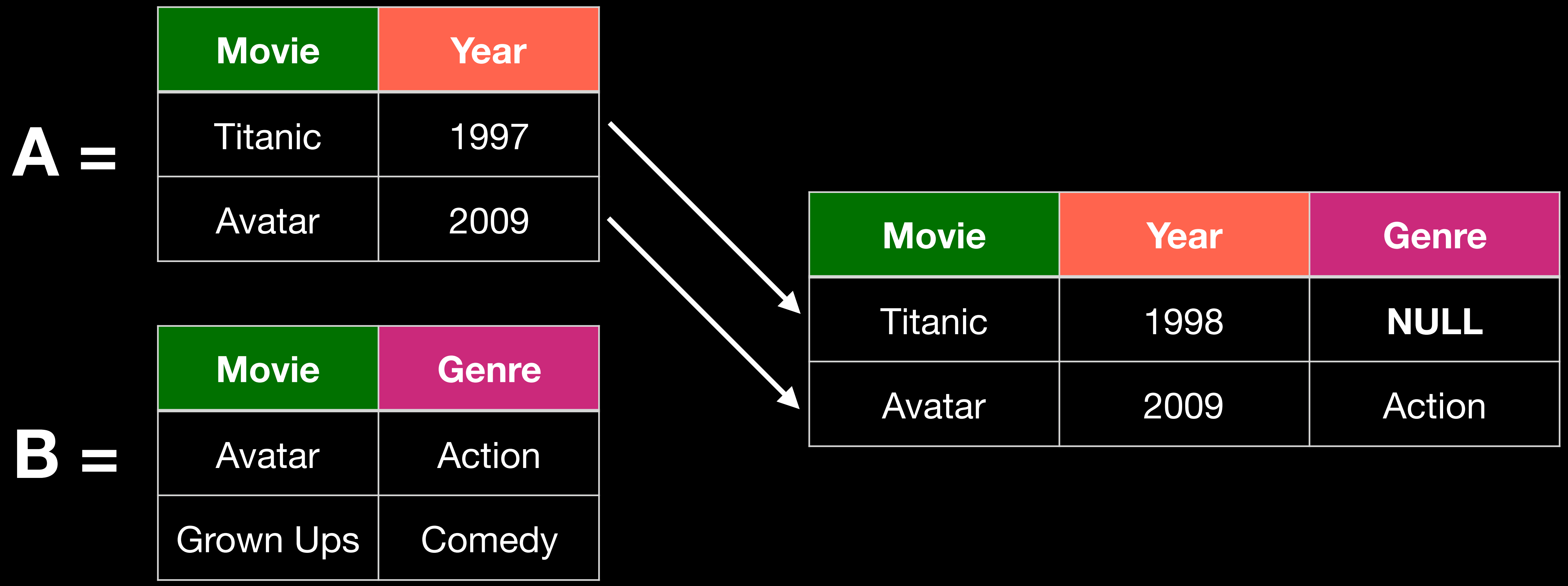
Movie	Year
Titanic	1997
Avatar	2009

B =

Movie	Genre
Avatar	Action
Grown Ups	Comedy

```
SELECT …
FROM   A LEFT JOIN B
ON     A.movie = B.movie
WHERE  …
```

Left Join



A =

Movie	Year
Titanic	1997
Avatar	2009

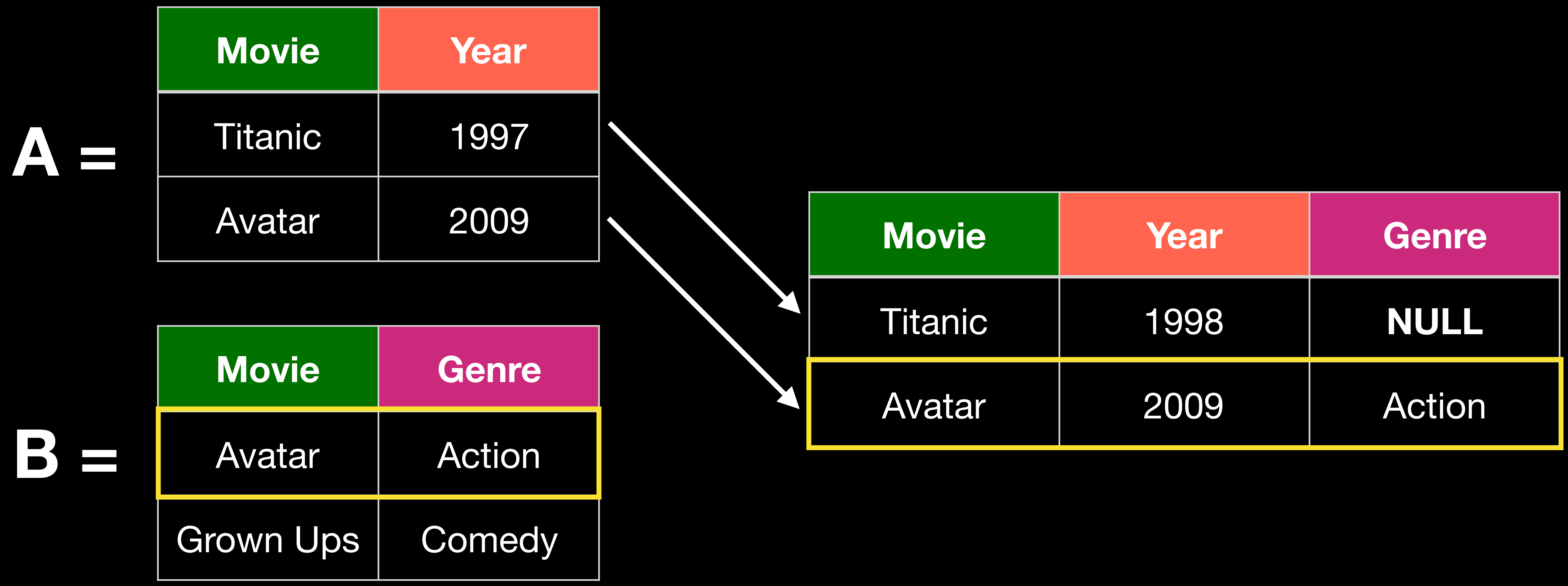
B =

Movie	Genre
Avatar	Action
Grown Ups	Comedy

Movie	Year	Genre
Titanic	1998	**NULL**
Avatar	2009	Action

```
SELECT …
FROM   A LEFT JOIN B
ON     A.movie = B.movie
WHERE  …
```

Left Join



A =

Movie	Year
Titanic	1997
Avatar	2009

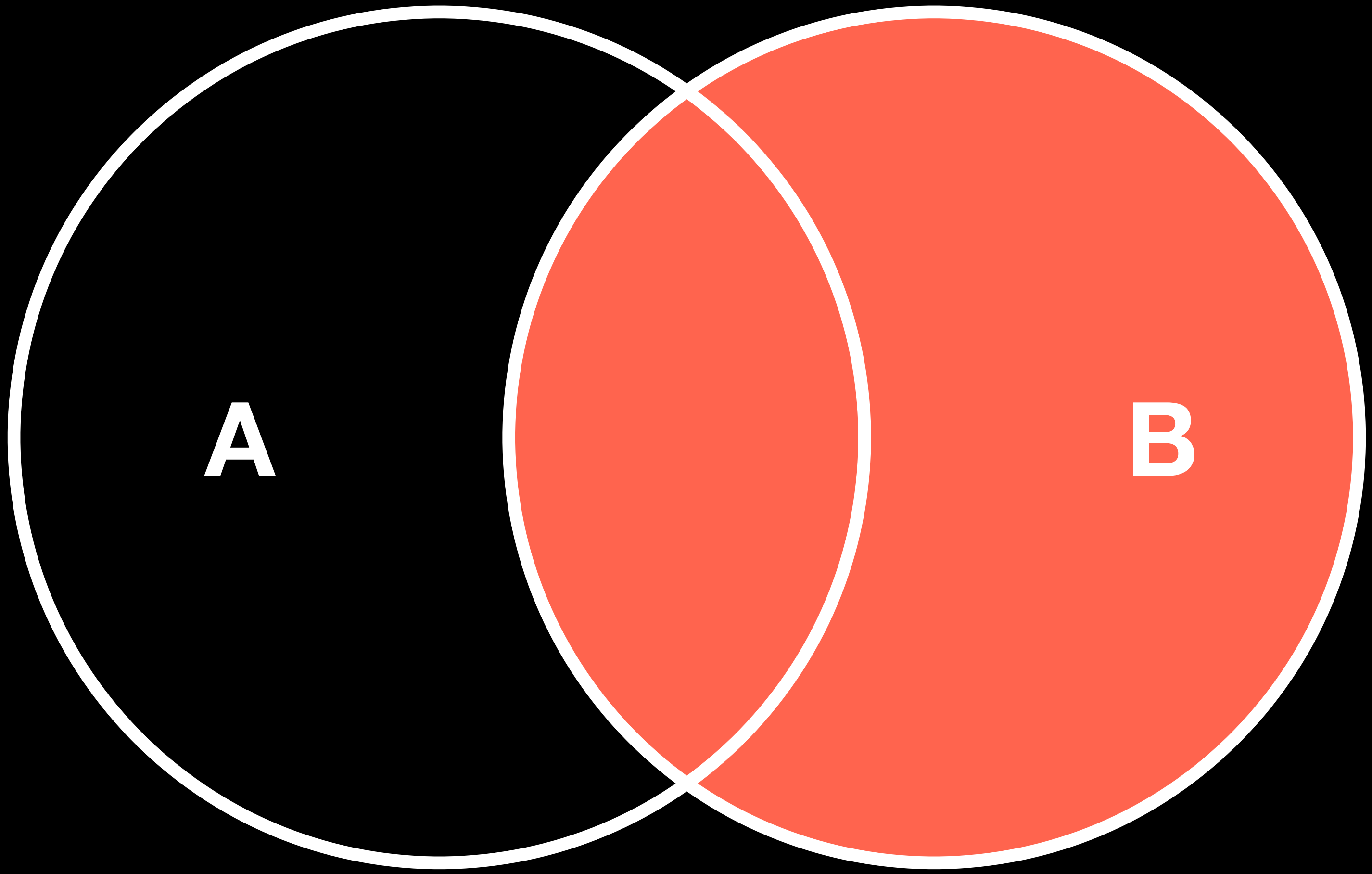
B =

Movie	Genre
Avatar	Action
Grown Ups	Comedy

Movie	Year	Genre
Titanic	1998	**NULL**
Avatar	2009	Action

```
SELECT …
FROM   A LEFT JOIN B
ON     A.movie = B.movie
WHERE  …
```

Right Join



```
SELECT [some attribute or column]
FROM   A LEFT JOIN B
ON     A.key = B.key
WHERE  [some condition is true]
```

Right Join

A =

Movie	Year
Titanic	1997
Avatar	2009

B =

Movie	Genre
Avatar	Action
Grown Ups	Comedy

```
SELECT …
FROM   A RIGHT JOIN B
ON     A.movie = B.movie
WHERE  …
```

Right Join

A =

Movie	Year
Titanic	1997
Avatar	2009

B =

Movie	Genre
Avatar	Action
Grown Ups	Comedy

```
SELECT …
FROM    A RIGHT JOIN B
ON      A.movie = B.movie
WHERE   …
```

Right Join

A =

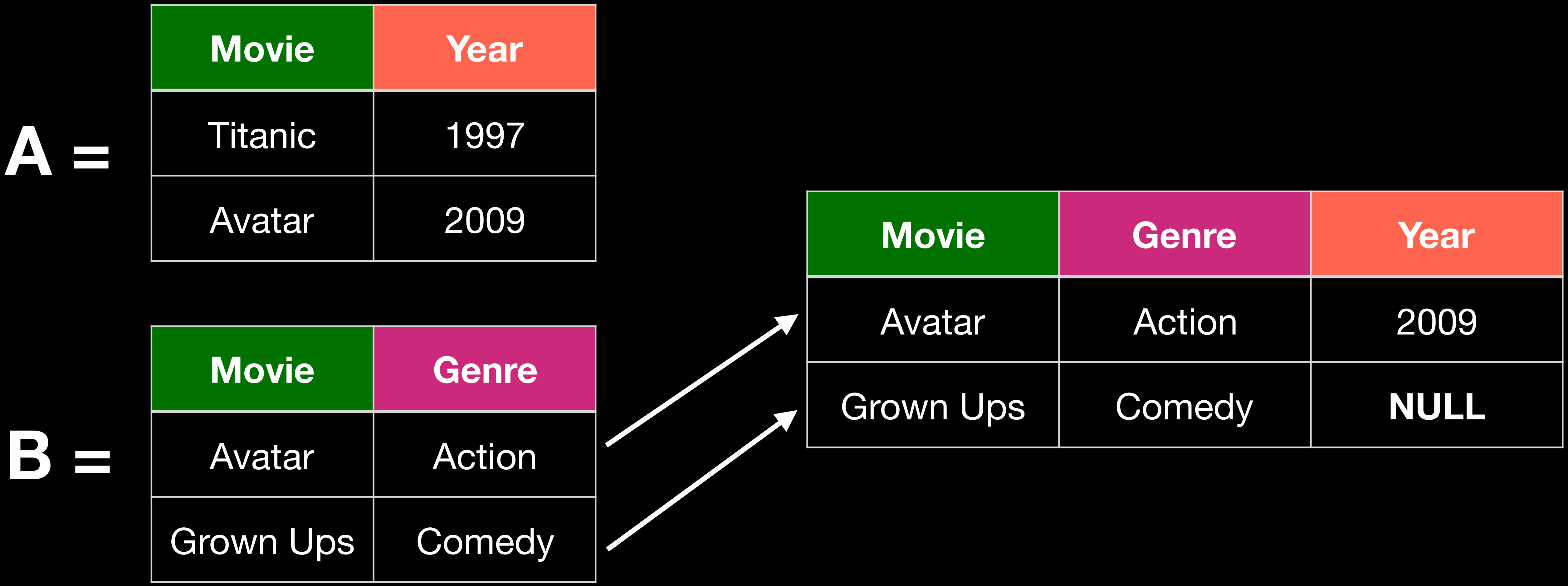
Movie	Year
Titanic	1997
Avatar	2009

B =

Movie	Genre
Avatar	Action
Grown Ups	Comedy

```
SELECT …
FROM   A RIGHT JOIN B
ON     A.movie = B.movie
WHERE  …
```

Right Join



A =

Movie	Year
Titanic	1997
Avatar	2009

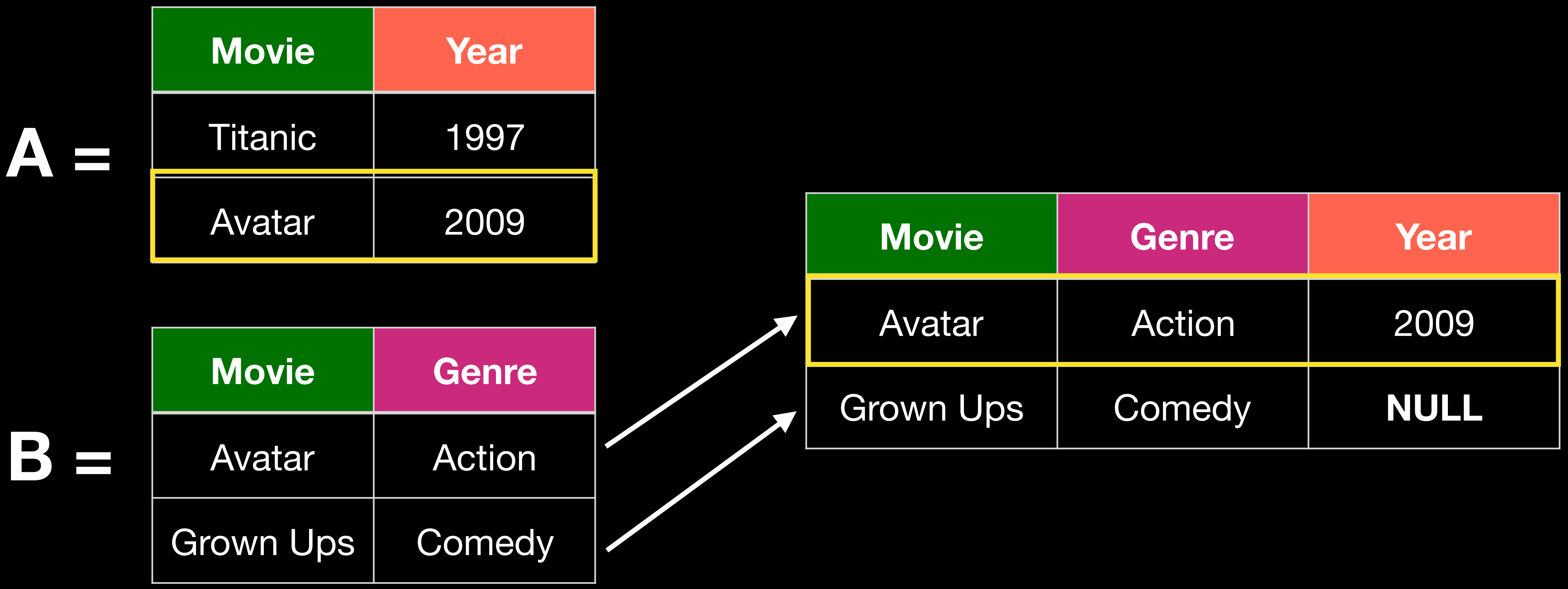
B =

Movie	Genre
Avatar	Action
Grown Ups	Comedy

Movie	Genre	Year
Avatar	Action	2009
Grown Ups	Comedy	**NULL**

```
SELECT …
FROM    A RIGHT JOIN B
ON      A.movie = B.movie
WHERE   …
```

Right Join



A =

Movie	Year
Titanic	1997
Avatar	2009

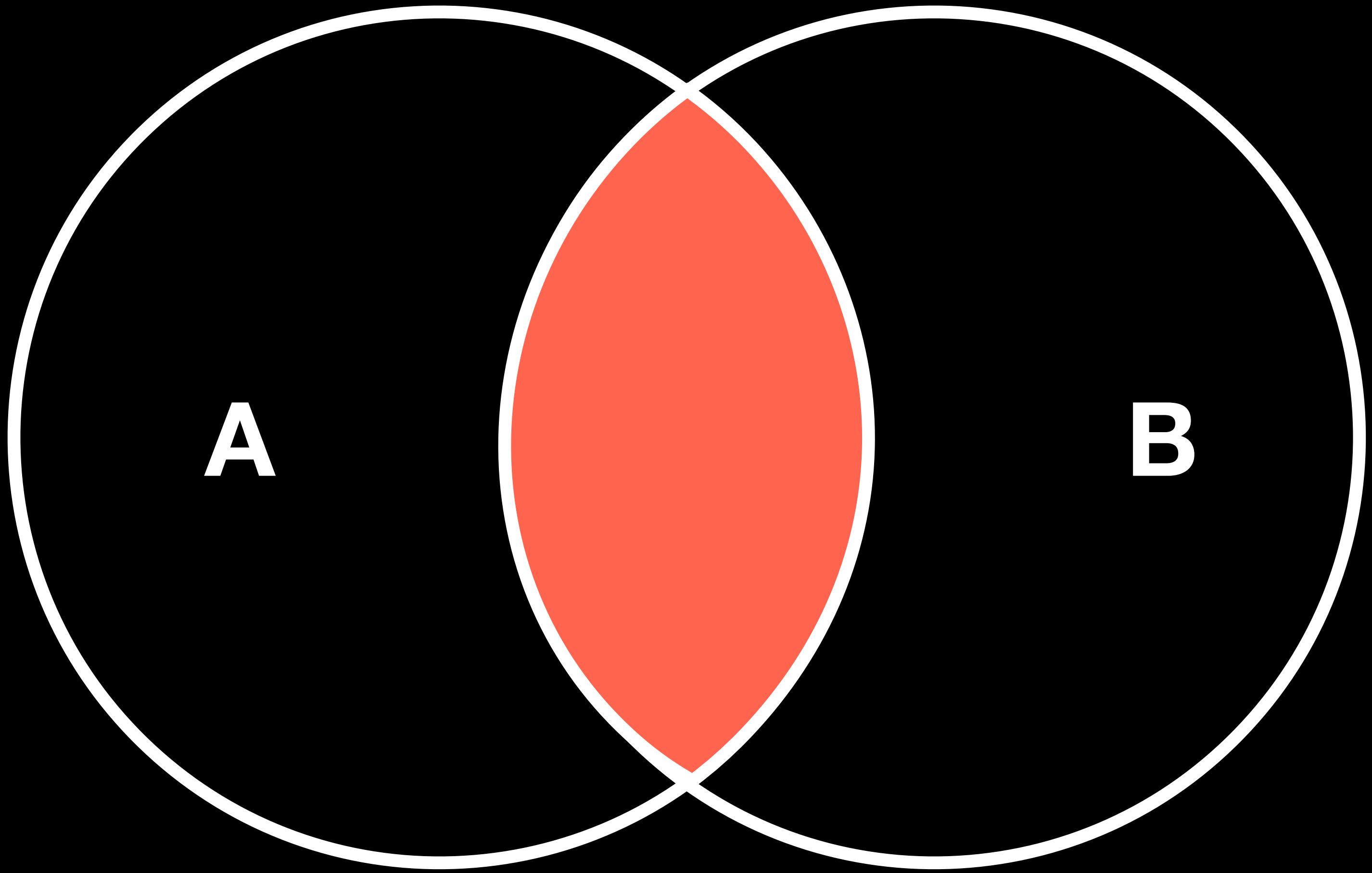
B =

Movie	Genre
Avatar	Action
Grown Ups	Comedy

Movie	Genre	Year
Avatar	Action	2009
Grown Ups	Comedy	**NULL**

```
SELECT  …
FROM    A RIGHT JOIN B
ON      A.movie = B.movie
WHERE   …
```

Inner Join



```
SELECT [some attribute or column]
FROM   A INNER JOIN B
ON     A.key = B.key
WHERE  [some condition is true]
```

Inner Join

A =

Movie	Year
Titanic	1997
Avatar	2009

B =

Movie	Genre
Avatar	Action
Grown Ups	Comedy

```
SELECT …
FROM   A INNER JOIN B
ON     A.movie = B.movie
WHERE  …
```

Inner Join

A =

Movie	Year
Titanic	1997
Avatar	2009

B =

Movie	Genre
Avatar	Action
Grown Ups	Comedy

```
SELECT …
FROM   A INNER JOIN B
ON     A.movie = B.movie
WHERE  …
```

Inner Join

A =

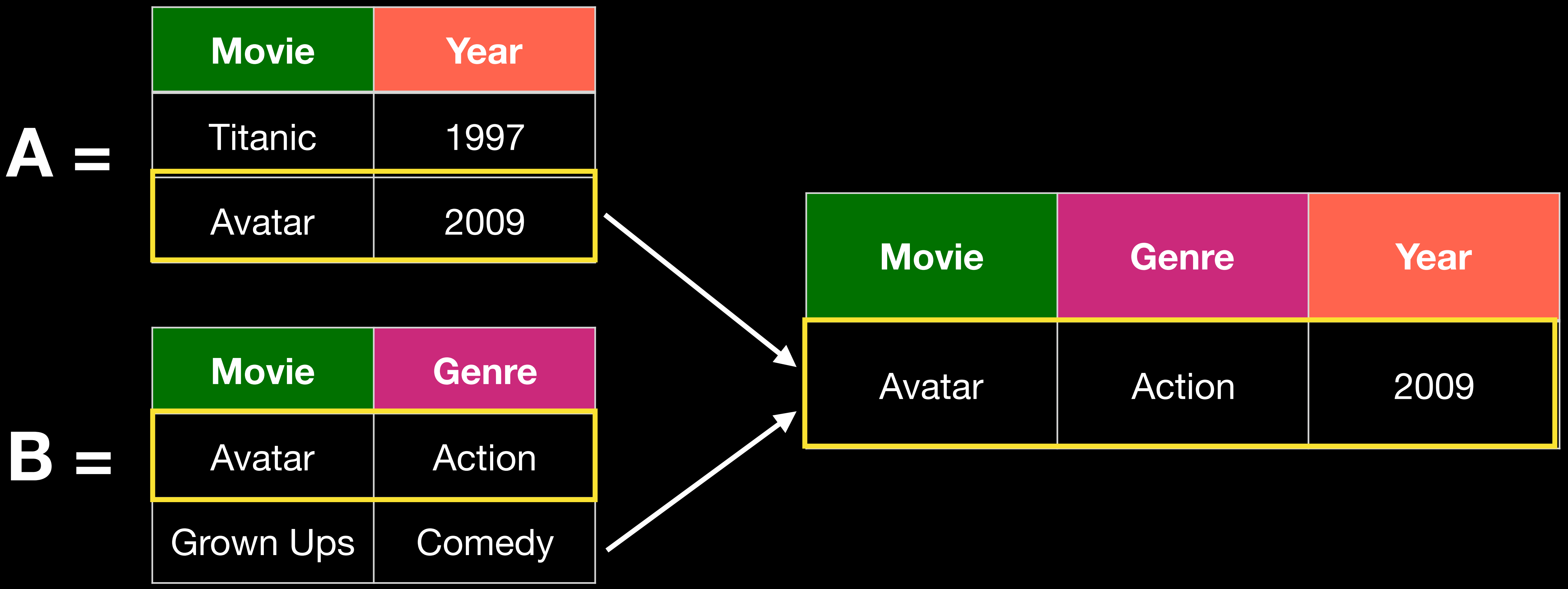
Movie	Year
Titanic	1997
Avatar	2009

B =

Movie	Genre
Avatar	Action
Grown Ups	Comedy

```
SELECT ...
FROM   A INNER JOIN B
ON     A.movie = B.movie
WHERE  ...
```

Inner Join



A =

Movie	Year
Titanic	1997
Avatar	2009

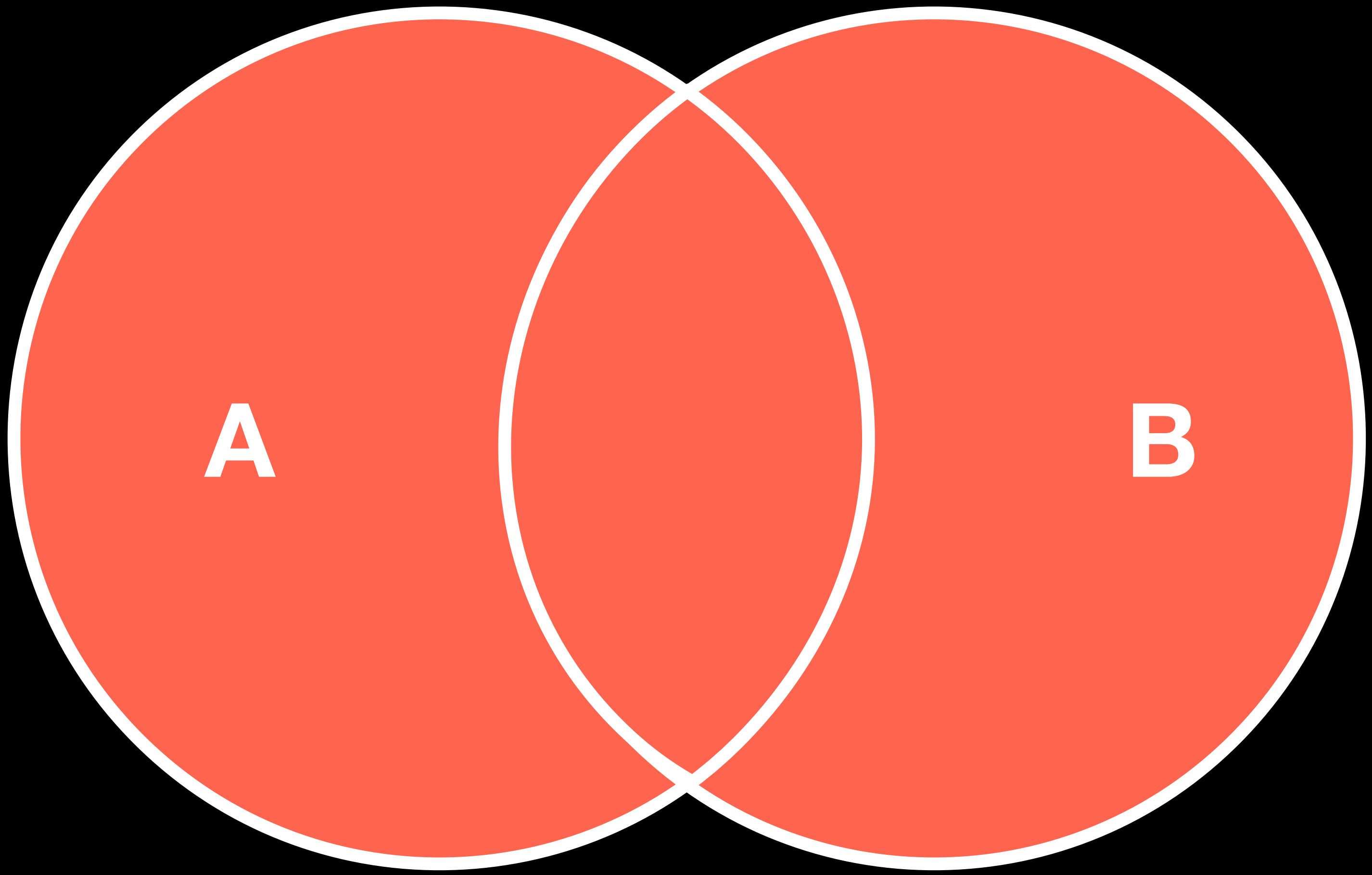
B =

Movie	Genre
Avatar	Action
Grown Ups	Comedy

Movie	Genre	Year
Avatar	Action	2009

```
SELECT …
FROM    A INNER JOIN B
ON      A.movie = B.movie
WHERE   …
```

Full Outer Join



```
SELECT [some attribute or column]
FROM   A FULL OUTER JOIN B
ON     A.key = B.key
WHERE  [some condition is true]
```

Full Outer Join

A =

Movie	Year
Titanic	1997
Avatar	2009

B =

Movie	Genre
Avatar	Action
Grown Ups	Comedy

```
SELECT …
FROM    A FULL OUTER JOIN B
ON      A.movie = B.movie
WHERE   …
```

Full Outer Join

A =

Movie	Year
Titanic	1997
Avatar	2009

B =

Movie	Genre
Avatar	Action
Grown Ups	Comedy

```
SELECT ...
FROM   A FULL OUTER JOIN B
ON     A.movie = B.movie
WHERE  ...
```

Full Outer Join

A =

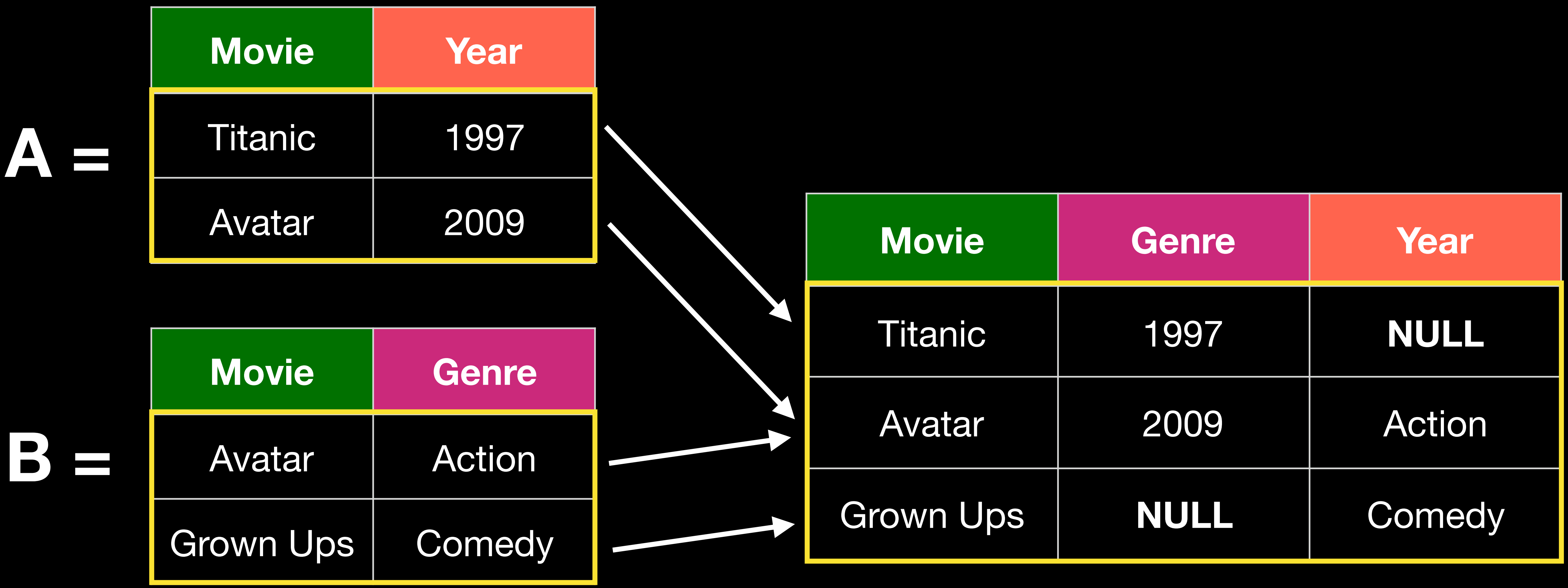
Movie	Year
Titanic	1997
Avatar	2009

B =

Movie	Genre
Avatar	Action
Grown Ups	Comedy

```
SELECT  …
FROM    A FULL OUTER JOIN B
ON      A.movie = B.movie
WHERE   …
```

Full Outer Join



A =

Movie	Year
Titanic	1997
Avatar	2009

B =

Movie	Genre
Avatar	Action
Grown Ups	Comedy

Movie	Genre	Year
Titanic	1997	**NULL**
Avatar	2009	Action
Grown Ups	**NULL**	Comedy

```
SELECT …
FROM   A FULL OUTER JOIN B
ON     A.movie = B.movie
WHERE  …
```

QUESTIONS?

LET ME KNOW!