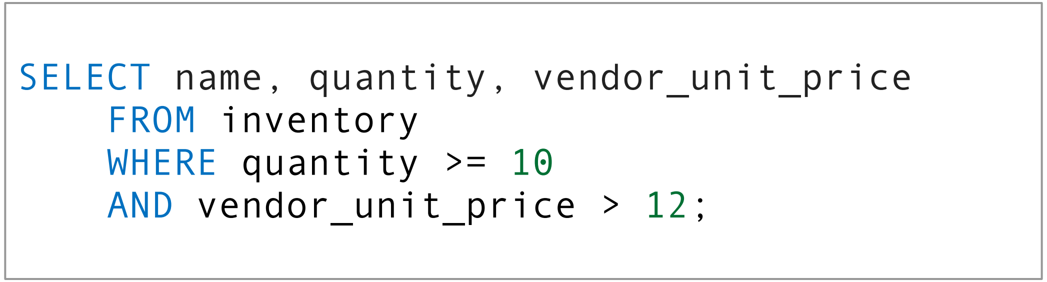
METIS

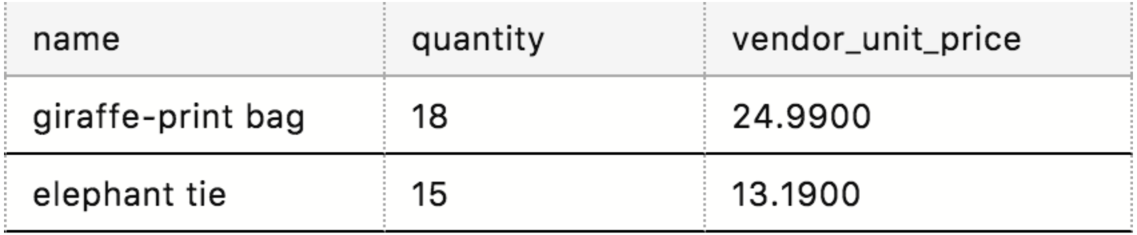
Intro to SQL

Joins

Querying and Filtering



```sql
SELECT name, quantity, vendor_unit_price
    FROM inventory
    WHERE quantity >= 10
    AND vendor_unit_price > 12;
```



name	quantity	vendor_unit_price
giraffe-print bag	18	24.9900
elephant tie	15	13.1900

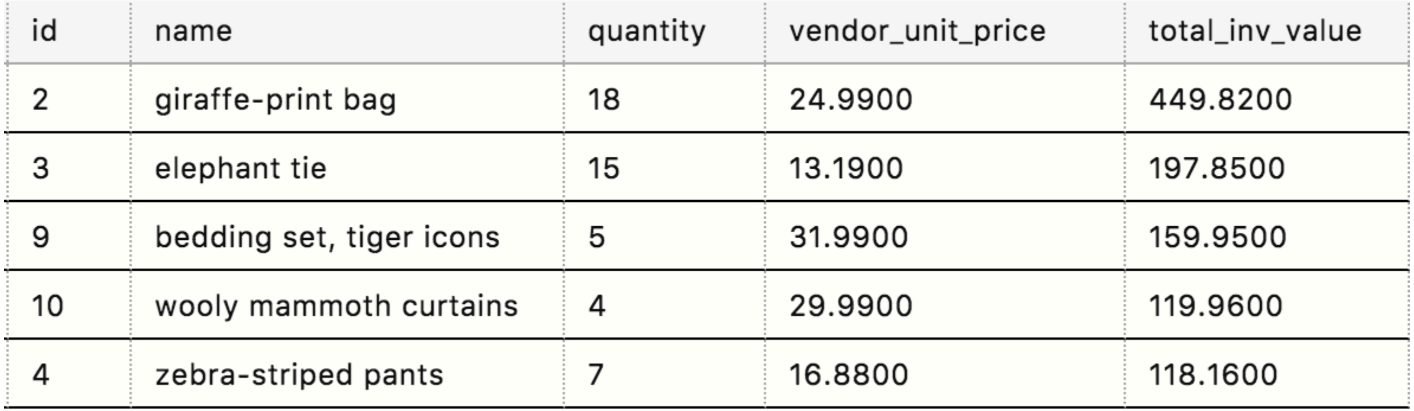
Querying and Conditioning

VALUE	Vendor Unit Price
High	More than $20
Medium	Between $10 and $20
Low	$10 or Less

```sql
SELECT name, vendor_unit_price,
    CASE
         WHEN vendor_unit_price IS NULL THEN NULL
         WHEN vendor_unit_price > 20 THEN “high”
         WHEN vendor_unit_price > 10 THEN “medium”
         ELSE “low”
    END AS value_class
FROM inventory;
```

Selecting Top Values, Aliasing

```
SELECT TOP 5 id, name, quantity, vendor_unit_price,
    quantity*vendor_unit_price AS total_inv_value
    FROM inventory
    ORDER BY total_inv_value DESC;
```



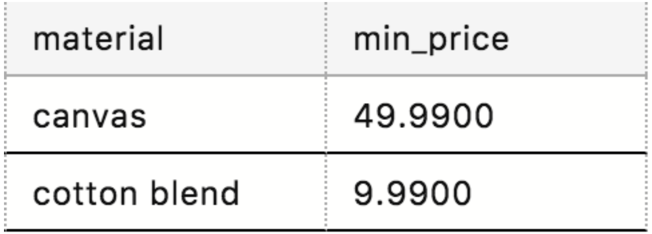
id	name	quantity	vendor_unit_price	total_inv_value
2	giraffe-print bag	18	24.9900	449.8200
3	elephant tie	15	13.1900	197.8500
9	bedding set, tiger icons	5	31.9900	159.9500
10	wooly mammoth curtains	4	29.9900	119.9600
4	zebra-striped pants	7	16.8800	118.1600

Aggregating, Filtering, Wildcards

Question:

What is the least expensive item made from each material that starts with the letter ‘c’?

```
SELECT material,
    MIN(sales_price) AS min_price
    FROM item_details
    GROUP BY material
    HAVING material LIKE ‘c%’;
```



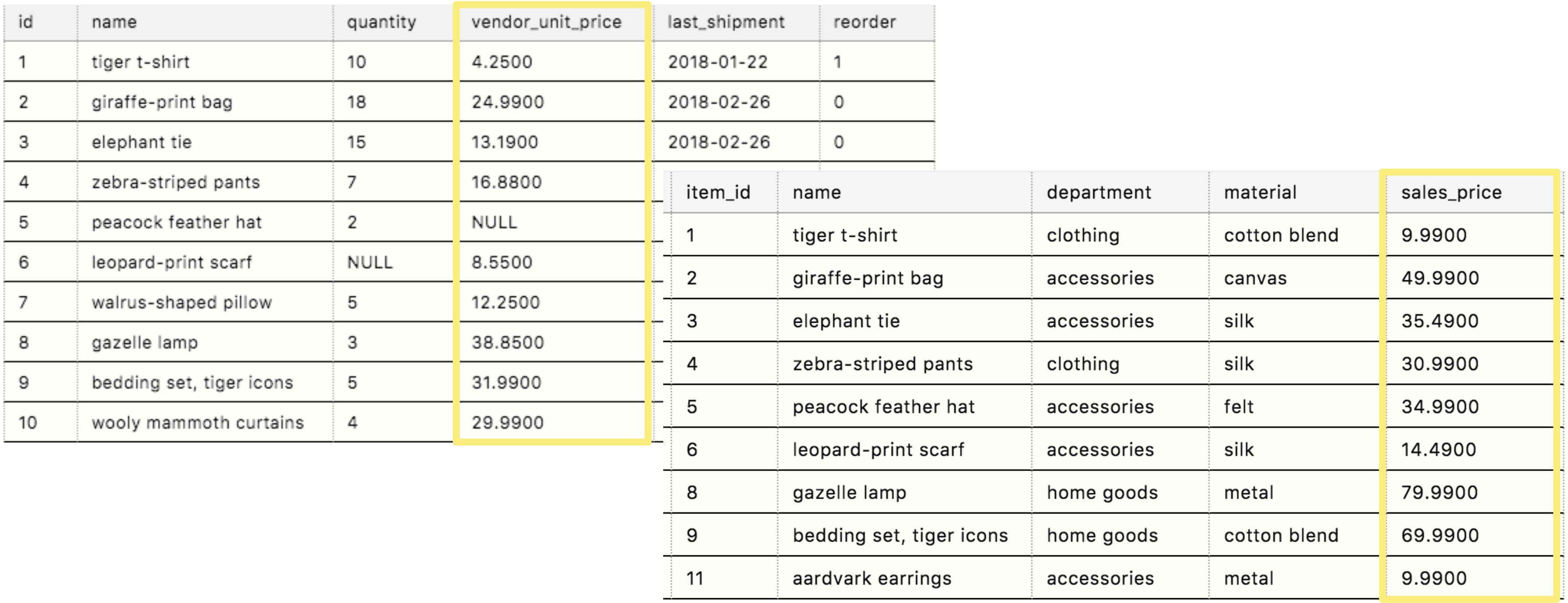
material	min_price
canvas	49.9900
cotton blend	9.9900

JOINS I:
CROSS JOIN

METIS

Joining Data Tables

Question:
How much markup does the company charge for each item in stock?

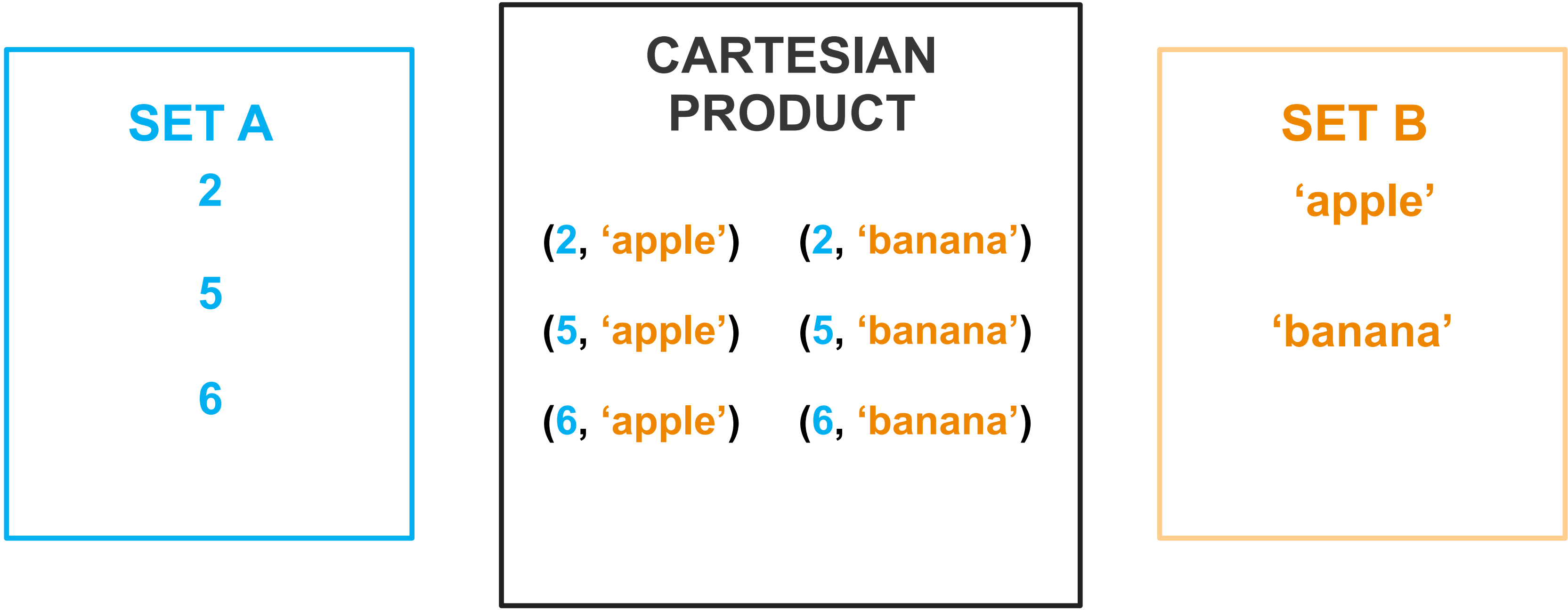


id	name	quantity	vendor_unit_price	last_shipment	reorder
1	tiger t-shirt	10	4.2500	2018-01-22	1
2	giraffe-print bag	18	24.9900	2018-02-26	0
3	elephant tie	15	13.1900	2018-02-26	0
4	zebra-striped pants	7	16.8800		
5	peacock feather hat	2	NULL		
6	leopard-print scarf	NULL	8.5500		
7	walrus-shaped pillow	5	12.2500		
8	gazelle lamp	3	38.8500		
9	bedding set, tiger icons	5	31.9900		
10	wooly mammoth curtains	4	29.9900		

item_id	name	department	material	sales_price
1	tiger t-shirt	clothing	cotton blend	9.9900
2	giraffe-print bag	accessories	canvas	49.9900
3	elephant tie	accessories	silk	35.4900
4	zebra-striped pants	clothing	silk	30.9900
5	peacock feather hat	accessories	felt	34.9900
6	leopard-print scarf	accessories	silk	14.4900
8	gazelle lamp	home goods	metal	79.9900
9	bedding set, tiger icons	home goods	cotton blend	69.9900
11	aardvark earrings	accessories	metal	9.9900

Cross Join

Cartesian Product - product of two sets; yields all possible ordered combinations (a, b)



SET A

2

5

6

CARTESIAN PRODUCT

(2, ‘apple’) (2, ‘banana’)

(5, ‘apple’) (5, ‘banana’)

(6, ‘apple’) (6, ‘banana’)

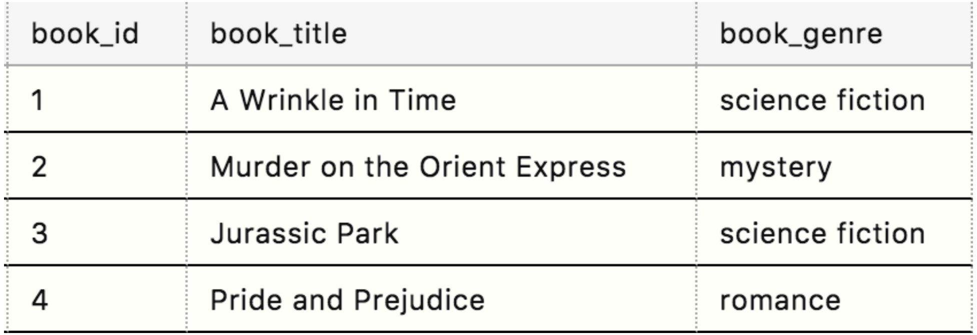
SET B

‘apple’

‘banana’

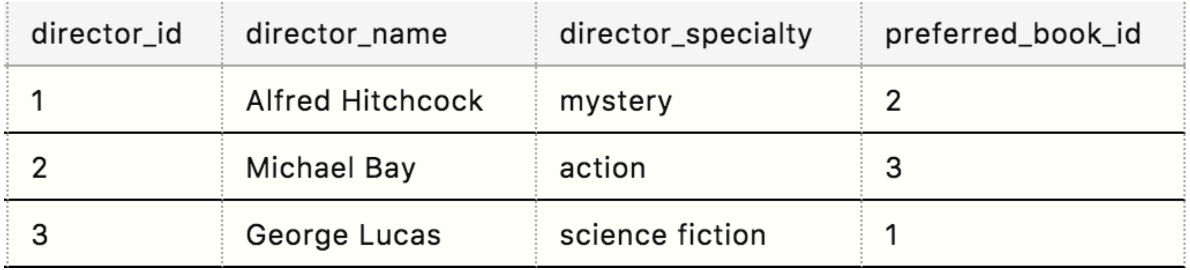
Cross Join Example

Books Table



book_id	book_title	book_genre
1	A Wrinkle in Time	science fiction
2	Murder on the Orient Express	mystery
3	Jurassic Park	science fiction
4	Pride and Prejudice	romance

Directors Table

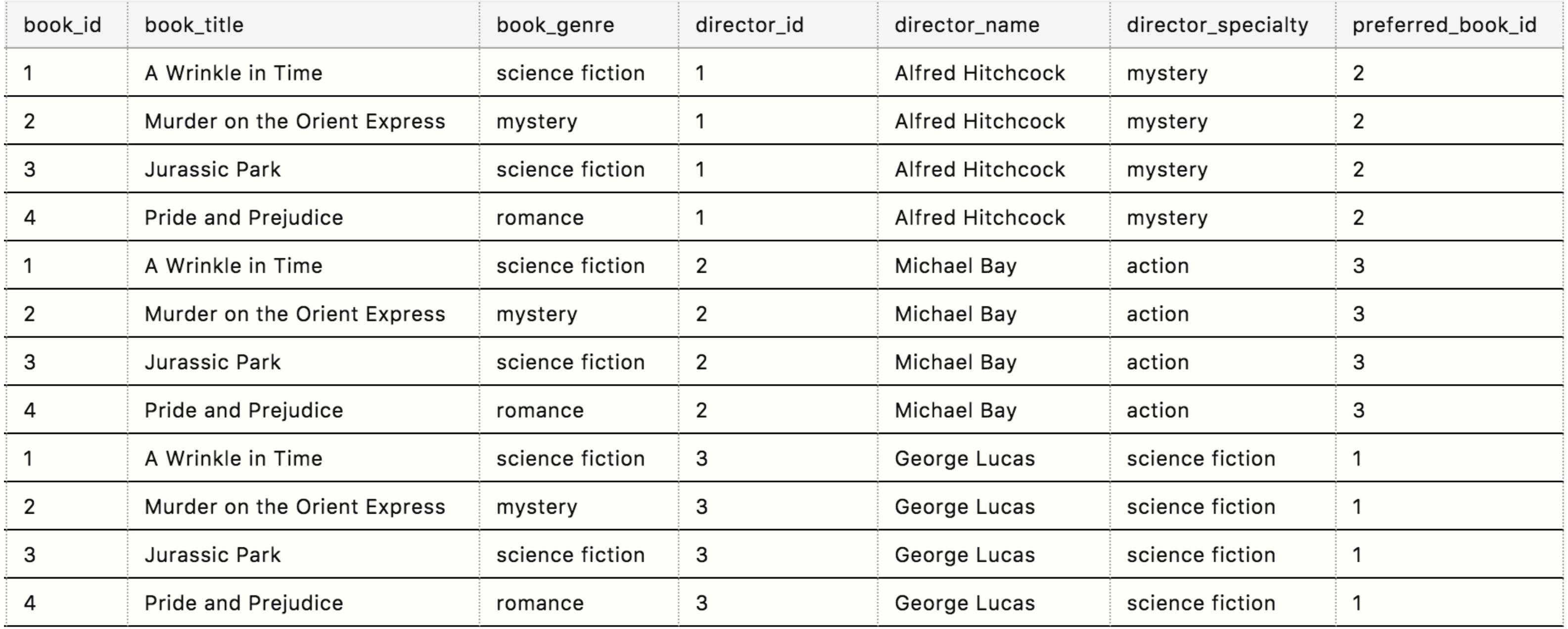


director_id	director_name	director_specialty	preferred_book_id
1	Alfred Hitchcock	mystery	2
2	Michael Bay	action	3
3	George Lucas	science fiction	1

Cross Join Example

Cartesian Product: Books x Directors

```sql
SELECT * FROM books, directors;
```



book_id	book_title	book_genre	director_id	director_name	director_specialty	preferred_book_id
1	A Wrinkle in Time	science fiction	1	Alfred Hitchcock	mystery	2
2	Murder on the Orient Express	mystery	1	Alfred Hitchcock	mystery	2
3	Jurassic Park	science fiction	1	Alfred Hitchcock	mystery	2
4	Pride and Prejudice	romance	1	Alfred Hitchcock	mystery	2
1	A Wrinkle in Time	science fiction	2	Michael Bay	action	3
2	Murder on the Orient Express	mystery	2	Michael Bay	action	3
3	Jurassic Park	science fiction	2	Michael Bay	action	3
4	Pride and Prejudice	romance	2	Michael Bay	action	3
1	A Wrinkle in Time	science fiction	3	George Lucas	science fiction	1
2	Murder on the Orient Express	mystery	3	George Lucas	science fiction	1
3	Jurassic Park	science fiction	3	George Lucas	science fiction	1
4	Pride and Prejudice	romance	3	George Lucas	science fiction	1

Note: Cartesian product is also called a CROSS JOIN.

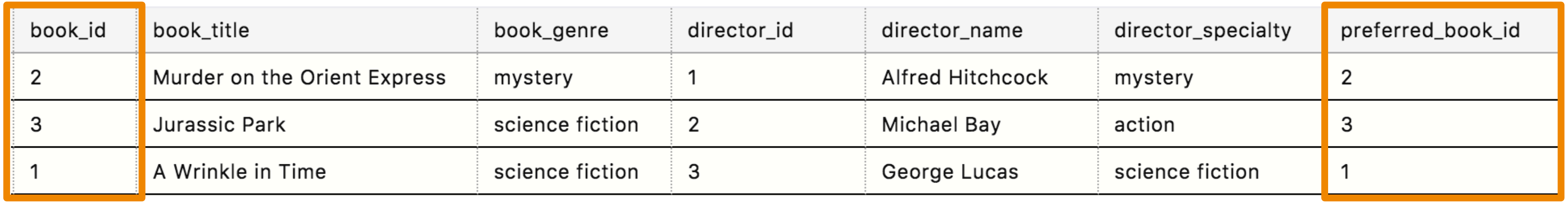
Cross Join Filtered

Question:

What are the titles of the books the directors would like to make into a movie?

```
SELECT * FROM books, directors
    WHERE book_id = preferred_book_id;
```

Implicit Inner Join



book_id	book_title	book_genre	director_id	director_name	director_specialty	preferred_book_id
2	Murder on the Orient Express	mystery	1	Alfred Hitchcock	mystery	2
3	Jurassic Park	science fiction	2	Michael Bay	action	3
1	A Wrinkle in Time	science fiction	3	George Lucas	science fiction	1

Note: Book IDs now match, but not sorted.
Cartesian product created first then rows eliminated with WHERE clause.

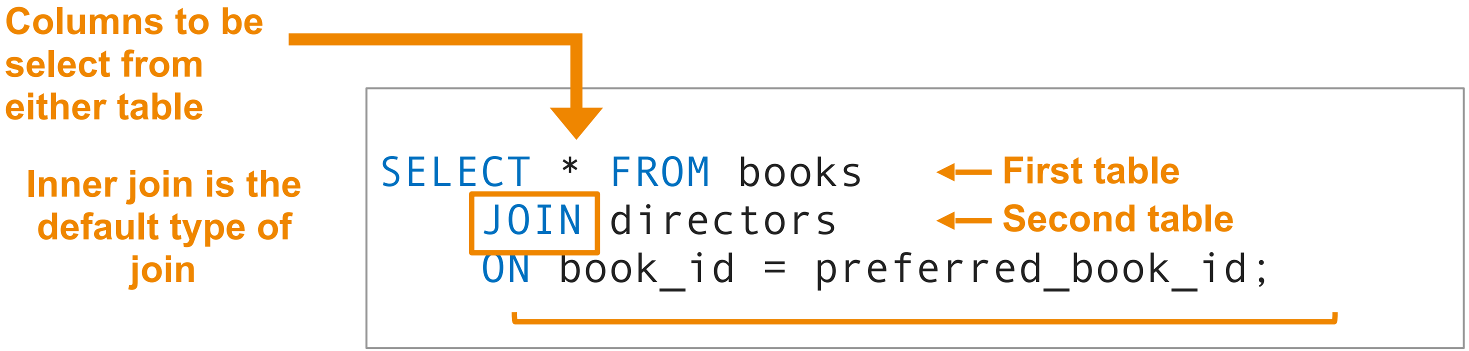
JOINS II:
INNER, OUTER, LEFT, RIGHT

METIS

Inner Join (Explicit)

Question:

What are the titles of the books the directors would like to make into a movie?



Columns to be select from either table

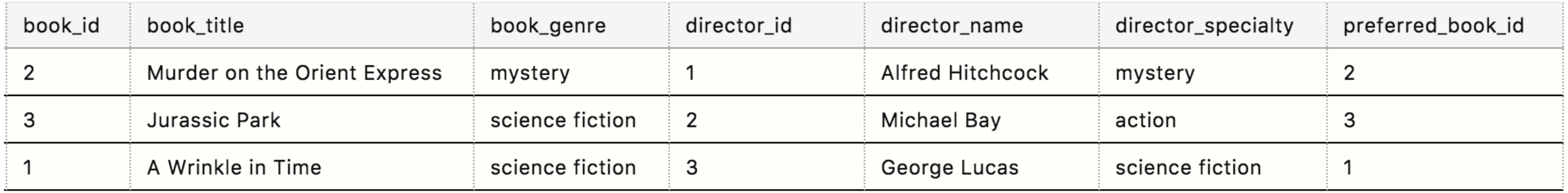
Inner join is the default type of join

```
SELECT * FROM books
    JOIN directors
    ON book_id = preferred_book_id;
```

First table

Second table

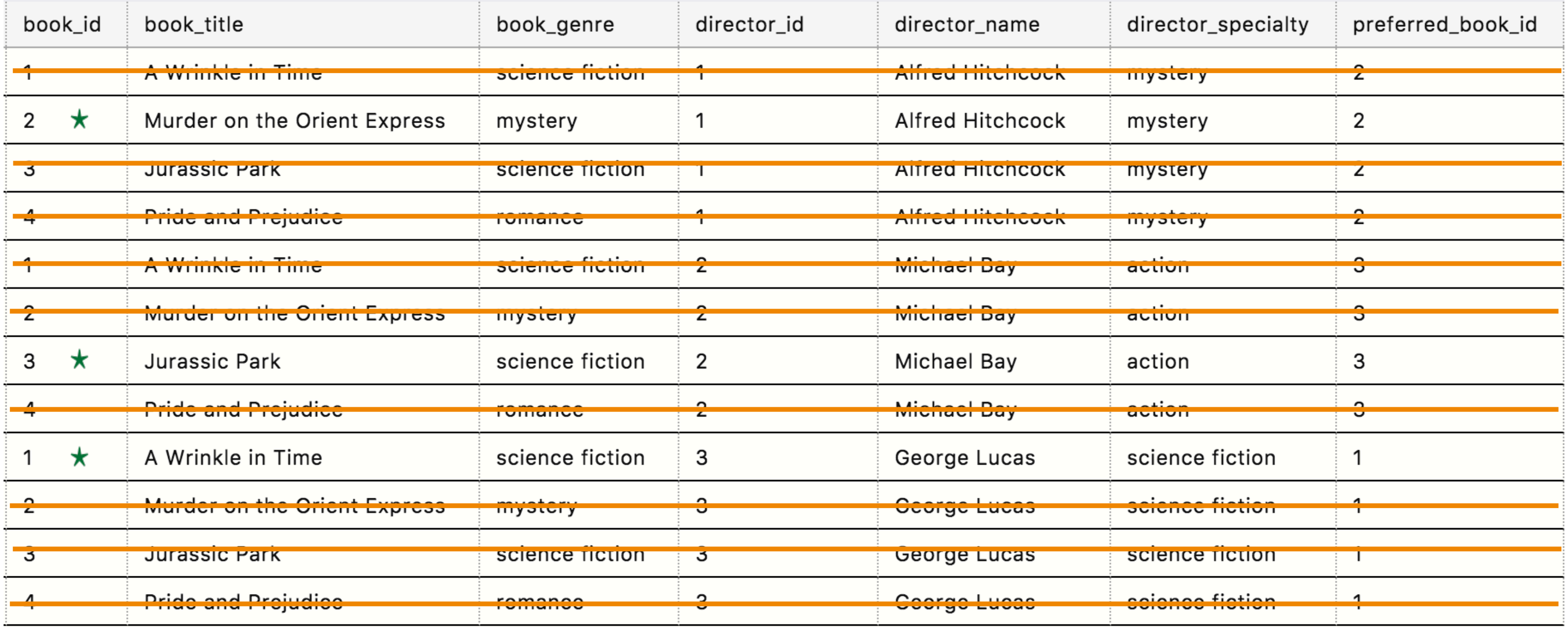
ON clause provides filtering; functions similar to WHERE



book_id	book_title	book_genre	director_id	director_name	director_specialty	preferred_book_id
2	Murder on the Orient Express	mystery	1	Alfred Hitchcock	mystery	2
3	Jurassic Park	science fiction	2	Michael Bay	action	3
1	A Wrinkle in Time	science fiction	3	George Lucas	science fiction	1

Inner Join

```
ON book_id = preferred_book_id
```

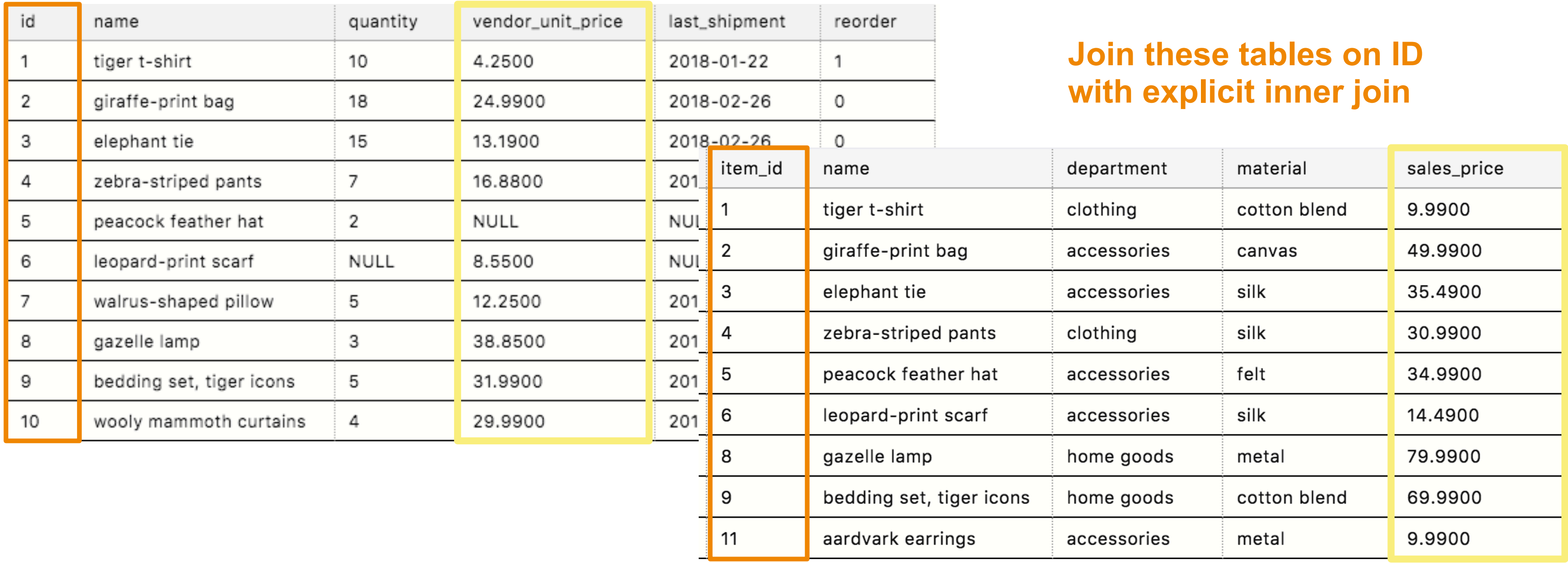


book_id	book_title	book_genre	director_id	director_name	director_specialty	preferred_book_id
~~1~~	~~A Wrinkle in Time~~	~~science fiction~~	~~1~~	~~Alfred Hitchcock~~	~~mystery~~	~~2~~
2 ★	Murder on the Orient Express	mystery	1	Alfred Hitchcock	mystery	2
~~3~~	~~Jurassic Park~~	~~science fiction~~	~~1~~	~~Alfred Hitchcock~~	~~mystery~~	~~2~~
~~4~~	~~Pride and Prejudice~~	~~romance~~	~~1~~	~~Alfred Hitchcock~~	~~mystery~~	~~2~~
~~1~~	~~A Wrinkle in Time~~	~~science fiction~~	~~2~~	~~Michael Bay~~	~~action~~	~~3~~
~~2~~	~~Murder on the Orient Express~~	~~mystery~~	~~2~~	~~Michael Bay~~	~~action~~	~~3~~
3 ★	Jurassic Park	science fiction	2	Michael Bay	action	3
~~4~~	~~Pride and Prejudice~~	~~romance~~	~~2~~	~~Michael Bay~~	~~action~~	~~3~~
1 ★	A Wrinkle in Time	science fiction	3	George Lucas	science fiction	1
~~2~~	~~Murder on the Orient Express~~	~~mystery~~	~~3~~	~~George Lucas~~	~~science fiction~~	~~1~~
~~3~~	~~Jurassic Park~~	~~science fiction~~	~~3~~	~~George Lucas~~	~~science fiction~~	~~1~~
~~4~~	~~Pride and Prejudice~~	~~romance~~	~~3~~	~~George Lucas~~	~~science fiction~~	~~1~~

Inner Join

Question:

How much markup does the animal items company charge for each item in stock?



id	name	quantity	vendor_unit_price	last_shipment	reorder
1	tiger t-shirt	10	4.2500	2018-01-22	1
2	giraffe-print bag	18	24.9900	2018-02-26	0
3	elephant tie	15	13.1900	2018-02-26	0
4	zebra-striped pants	7	16.8800	201	
5	peacock feather hat	2	NULL	NUL	
6	leopard-print scarf	NULL	8.5500	NUL	
7	walrus-shaped pillow	5	12.2500	201	
8	gazelle lamp	3	38.8500	201	
9	bedding set, tiger icons	5	31.9900	201	
10	wooly mammoth curtains	4	29.9900	201	

item_id	name	department	material	sales_price
1	tiger t-shirt	clothing	cotton blend	9.9900
2	giraffe-print bag	accessories	canvas	49.9900
3	elephant tie	accessories	silk	35.4900
4	zebra-striped pants	clothing	silk	30.9900
5	peacock feather hat	accessories	felt	34.9900
6	leopard-print scarf	accessories	silk	14.4900
8	gazelle lamp	home goods	metal	79.9900
9	bedding set, tiger icons	home goods	cotton blend	69.9900
11	aardvark earrings	accessories	metal	9.9900

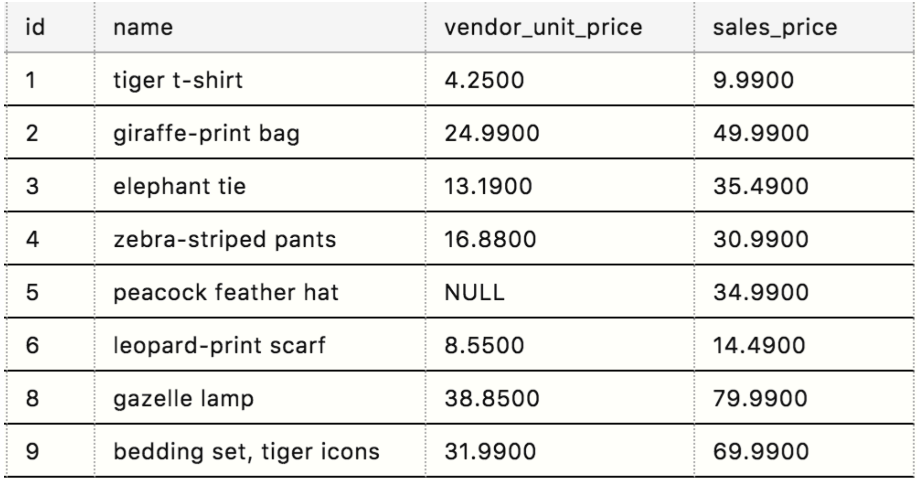
Join these tables on ID with explicit inner join

Inner Join

Explicitly tell SQL which table each column comes from: table.column

```
SELECT inventory.id, inventory.name,
       inventory.vendor_unit_price, item_details.sales_price
   FROM inventory
   INNER JOIN item_details
   ON inventory.id = item_details.item_id;
```

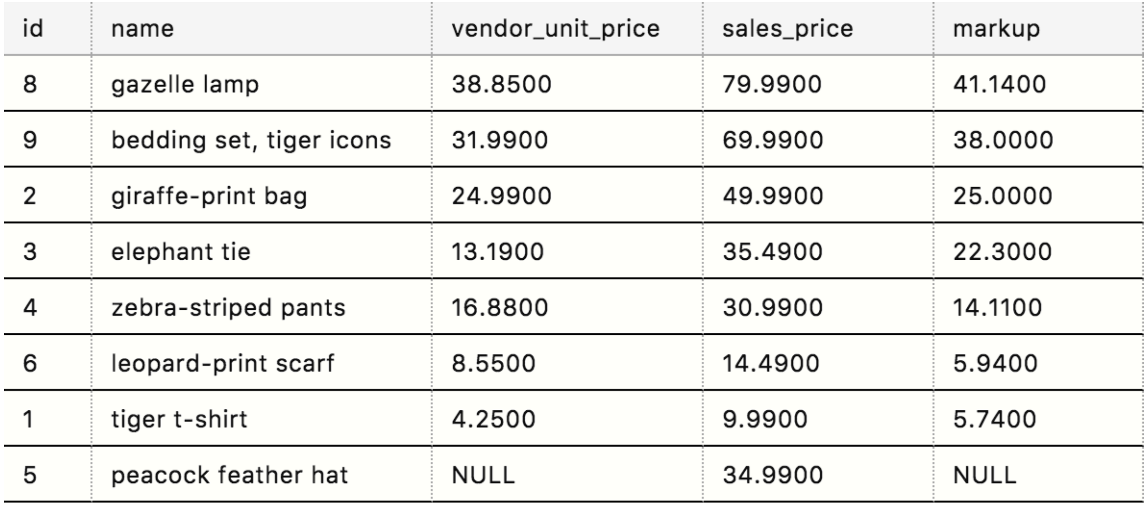
Explicitly stating INNER JOIN also acceptable



id	name	vendor_unit_price	sales_price
1	tiger t-shirt	4.2500	9.9900
2	giraffe-print bag	24.9900	49.9900
3	elephant tie	13.1900	35.4900
4	zebra-striped pants	16.8800	30.9900
5	peacock feather hat	NULL	34.9900
6	leopard-print scarf	8.5500	14.4900
8	gazelle lamp	38.8500	79.9900
9	bedding set, tiger icons	31.9900	69.9900

Inner Join

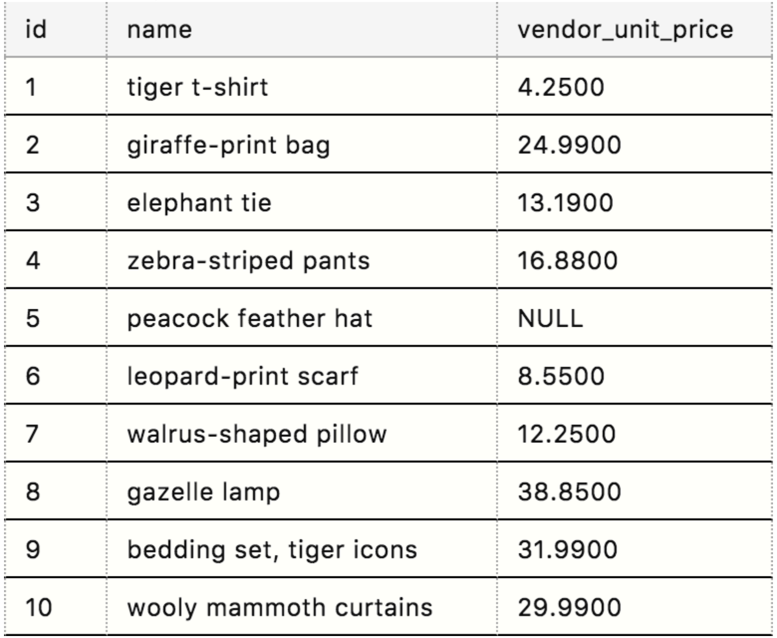
```
SELECT inventory.id, inventory.name,
       inventory.vendor_unit_price, item_details.sales_price,
       item_details.sales_price-inventory.vendor_unit_price AS markup
    FROM inventory
    INNER JOIN item_details
    ON inventory.id = item_details.item_id ORDER BY markup DESC;
```



id	name	vendor_unit_price	sales_price	markup
8	gazelle lamp	38.8500	79.9900	41.1400
9	bedding set, tiger icons	31.9900	69.9900	38.0000
2	giraffe-print bag	24.9900	49.9900	25.0000
3	elephant tie	13.1900	35.4900	22.3000
4	zebra-striped pants	16.8800	30.9900	14.1100
6	leopard-print scarf	8.5500	14.4900	5.9400
1	tiger t-shirt	4.2500	9.9900	5.7400
5	peacock feather hat	NULL	34.9900	NULL

Joining Tables

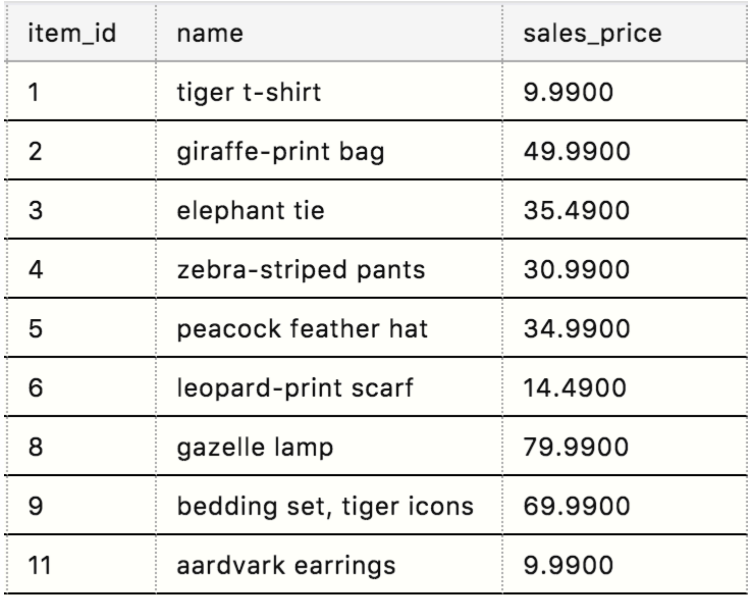
Inventory Table



id	name	vendor_unit_price
1	tiger t-shirt	4.2500
2	giraffe-print bag	24.9900
3	elephant tie	13.1900
4	zebra-striped pants	16.8800
5	peacock feather hat	NULL
6	leopard-print scarf	8.5500
7	walrus-shaped pillow	12.2500
8	gazelle lamp	38.8500
9	bedding set, tiger icons	31.9900
10	wooly mammoth curtains	29.9900

No information about item 11

Item Details Table



item_id	name	sales_price
1	tiger t-shirt	9.9900
2	giraffe-print bag	49.9900
3	elephant tie	35.4900
4	zebra-striped pants	30.9900
5	peacock feather hat	34.9900
6	leopard-print scarf	14.4900
8	gazelle lamp	79.9900
9	bedding set, tiger icons	69.9900
11	aardvark earrings	9.9900

No information about items 7 and 10

INNER JOIN must have information in both tables.
Drops NULL rows from Cartesian product when filtering with ON.

Outer Join

Unlike INNER JOIN, OUTER JOIN keeps rows where NULL appears in ON clause

Three types of OUTER JOINs to specify which NULL values to keep

1. **FULL OUTER JOIN** – keep everything
2. **LEFT OUTER JOIN** – keep all rows from “left” table,
   regardless if they appear in “right”
3. **RIGHT OUTER JOIN** – keep all rows from “right” table,
   regardless if they appear in “left”

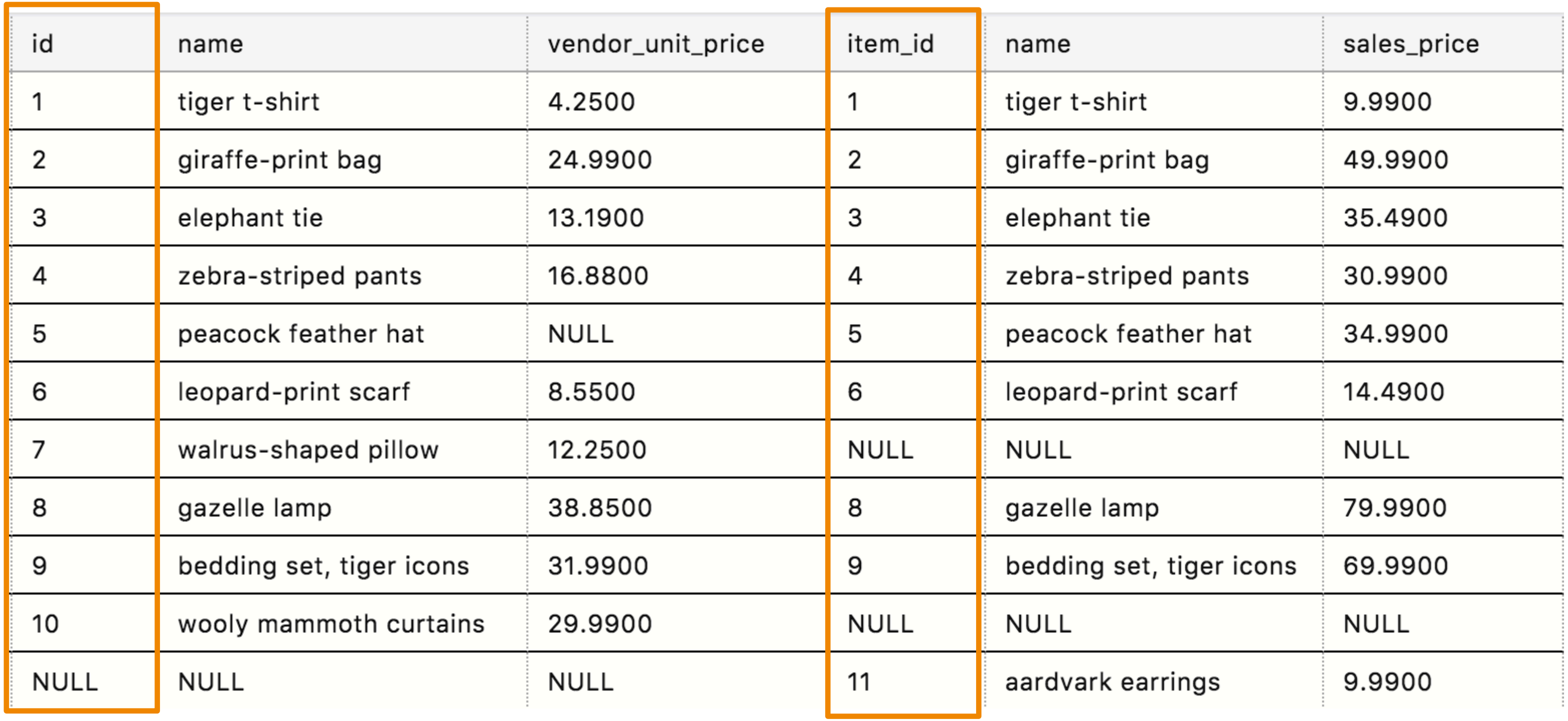
FULL OUTER JOIN

```
SELECT inventory.id, inventory.name,
       inventory.vendor_unit_price,
       item_details.item_id, item_details.name,
       item_details.sales_price
    FROM inventory
    FULL OUTER JOIN item_details
    ON inventory.id = item_details.item_id;
```

Only difference now: Specify FULL OUTER JOIN

FULL OUTER JOIN

NULL values appear in both columns from the ON statement



id	name	vendor_unit_price	item_id	name	sales_price
1	tiger t-shirt	4.2500	1	tiger t-shirt	9.9900
2	giraffe-print bag	24.9900	2	giraffe-print bag	49.9900
3	elephant tie	13.1900	3	elephant tie	35.4900
4	zebra-striped pants	16.8800	4	zebra-striped pants	30.9900
5	peacock feather hat	NULL	5	peacock feather hat	34.9900
6	leopard-print scarf	8.5500	6	leopard-print scarf	14.4900
7	walrus-shaped pillow	12.2500	NULL	NULL	NULL
8	gazelle lamp	38.8500	8	gazelle lamp	79.9900
9	bedding set, tiger icons	31.9900	9	bedding set, tiger icons	69.9900
10	wooly mammoth curtains	29.9900	NULL	NULL	NULL
NULL	NULL	NULL	11	aardvark earrings	9.9900

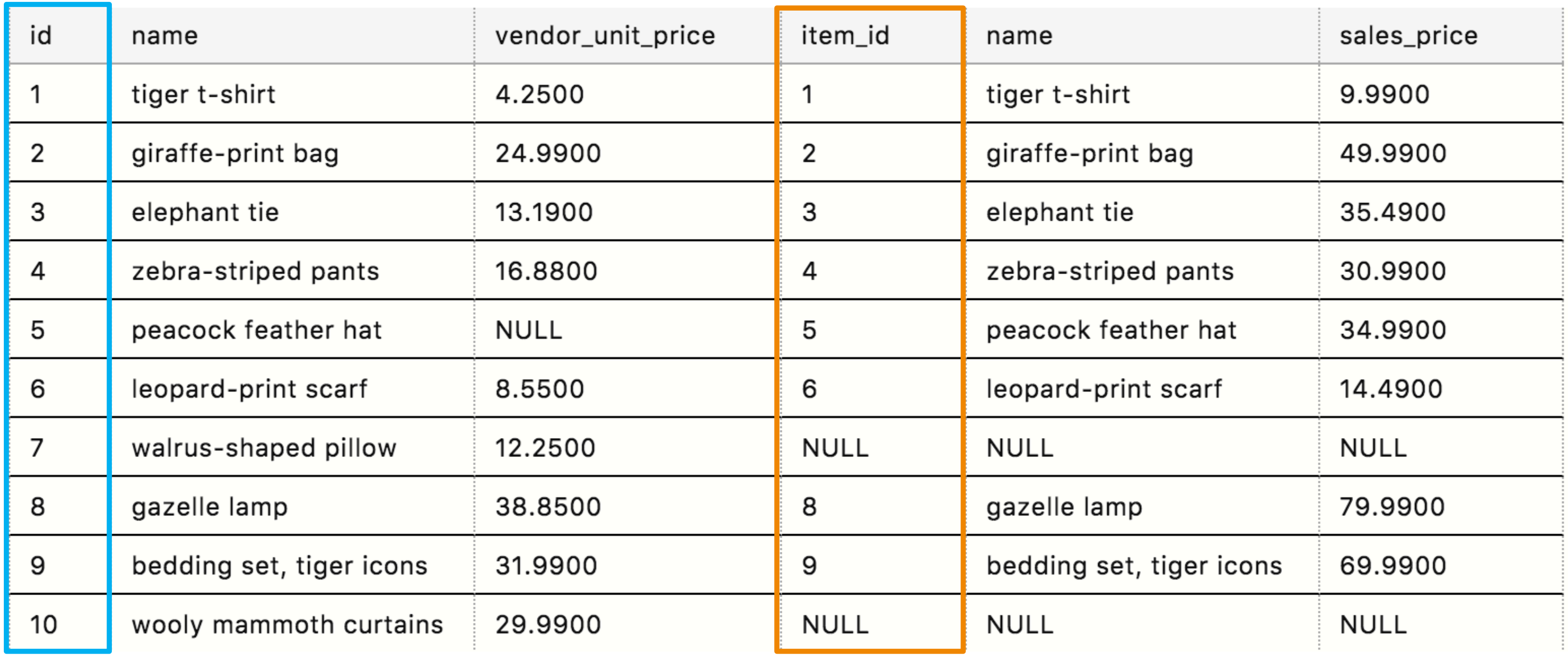
LEFT OUTER JOIN

```
SELECT inventory.id, inventory.name,
       inventory.vendor_unit_price,
       item_details.item_id, item_details.name,
       item_details.sales_price
    FROM inventory
    LEFT OUTER JOIN item_details
    ON inventory.id = item_details.item_id;
```

Left OUTER JOIN will require values for the “left” table (inventory)

LEFT OUTER JOIN

All data from “left” table is retained; NULL values allowed in the id column of the “right” table



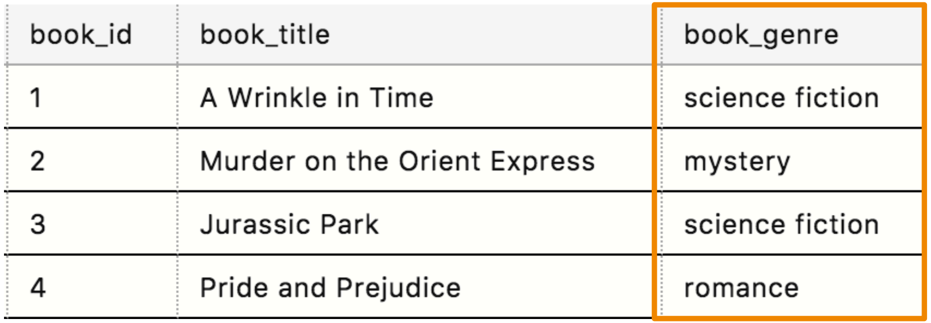
id	name	vendor_unit_price	item_id	name	sales_price
1	tiger t-shirt	4.2500	1	tiger t-shirt	9.9900
2	giraffe-print bag	24.9900	2	giraffe-print bag	49.9900
3	elephant tie	13.1900	3	elephant tie	35.4900
4	zebra-striped pants	16.8800	4	zebra-striped pants	30.9900
5	peacock feather hat	NULL	5	peacock feather hat	34.9900
6	leopard-print scarf	8.5500	6	leopard-print scarf	14.4900
7	walrus-shaped pillow	12.2500	NULL	NULL	NULL
8	gazelle lamp	38.8500	8	gazelle lamp	79.9900
9	bedding set, tiger icons	31.9900	9	bedding set, tiger icons	69.9900
10	wooly mammoth curtains	29.9900	NULL	NULL	NULL

Joining Not on Primary Keys

Question:

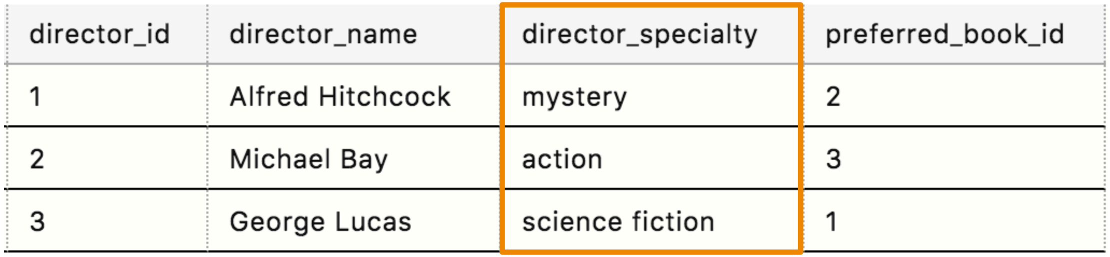
What possible book + director combinations align in genre-specialty?

Books Table



book_id	book_title	book_genre
1	A Wrinkle in Time	science fiction
2	Murder on the Orient Express	mystery
3	Jurassic Park	science fiction
4	Pride and Prejudice	romance

Directors Table



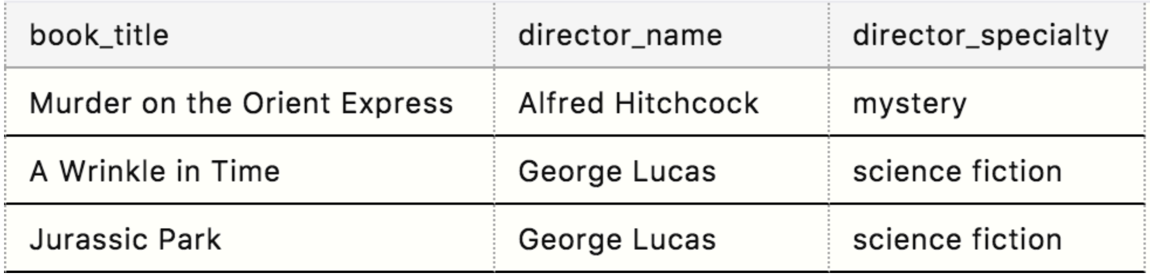
director_id	director_name	director_specialty	preferred_book_id
1	Alfred Hitchcock	mystery	2
2	Michael Bay	action	3
3	George Lucas	science fiction	1

Joining Not on Primary Keys

Question:

What possible book + director combinations align in genre-specialty?

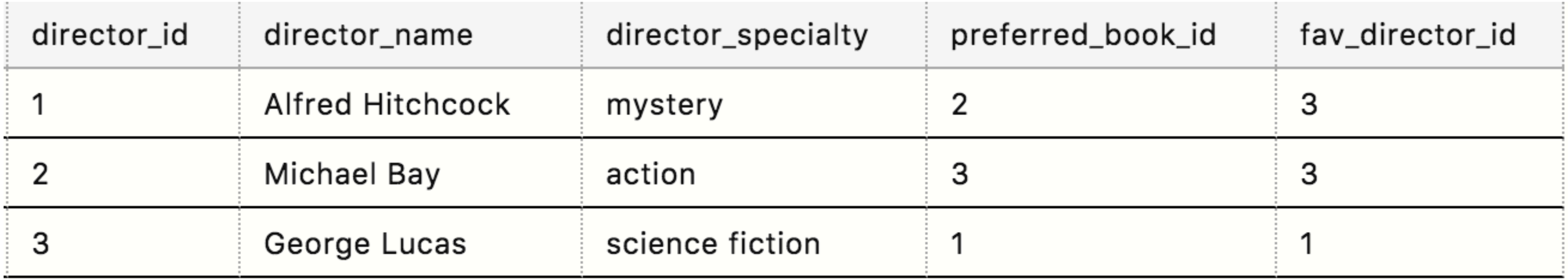
```
SELECT books.book_title, directors.director_name,
       directors.director_specialty
    FROM books JOIN directors
    ON books.book_genre = directors.director_specialty
    ORDER BY directors.director_name;
```



book_title	director_name	director_specialty
Murder on the Orient Express	Alfred Hitchcock	mystery
A Wrinkle in Time	George Lucas	science fiction
Jurassic Park	George Lucas	science fiction

Self-Join

Add one additional column to the Directors Table called “fav_director_id”



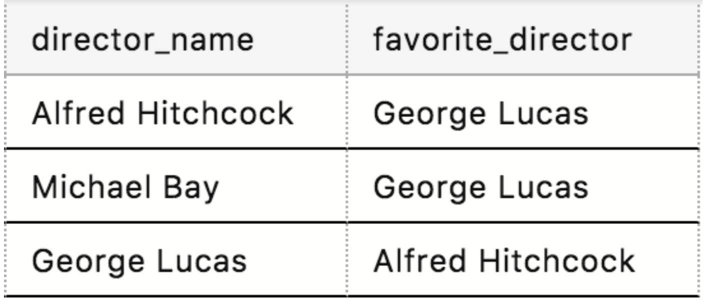
director_id	director_name	director_specialty	preferred_book_id	fav_director_id
1	Alfred Hitchcock	mystery	2	3
2	Michael Bay	action	3	3
3	George Lucas	science fiction	1	1

Question:
Which other director does each director in our table admire?

Self-Join

Question:
Which other director does each director in our table admire?

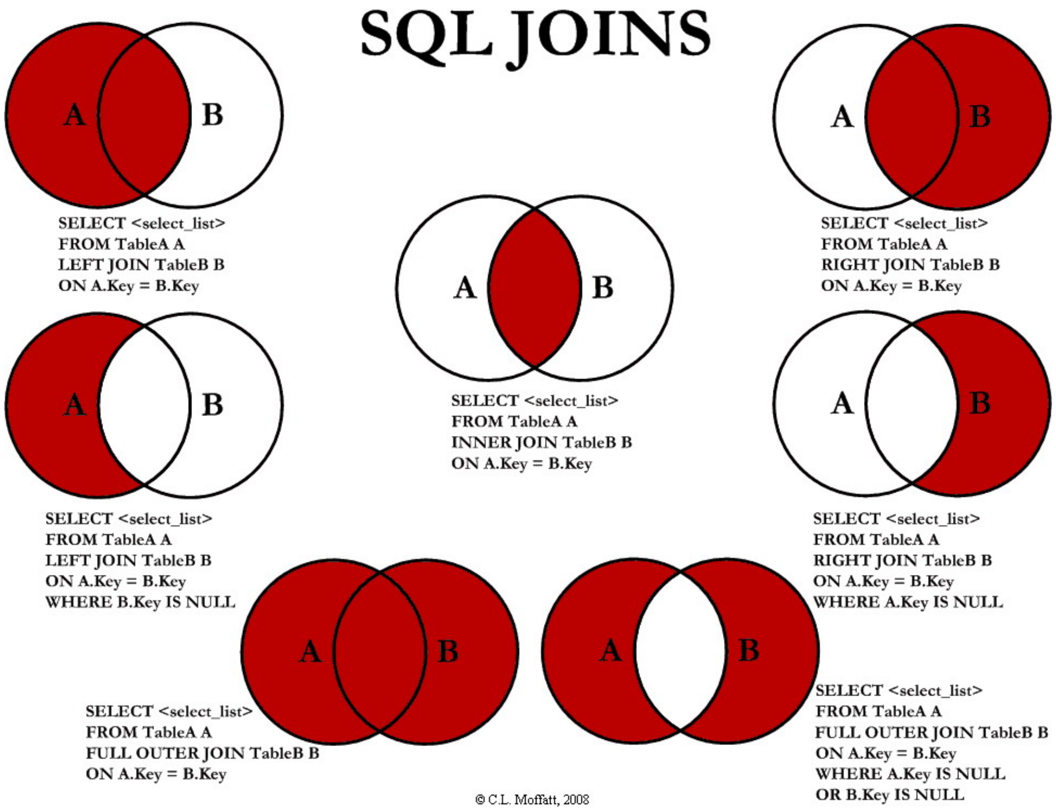
```
SELECT directors.director_name,
       favorite.director_name AS favorite_director
    FROM directors
    JOIN directors AS favorite
    ON directors.fav_director_id = favorite.director_id;
```



director_name	favorite_director
Alfred Hitchcock	George Lucas
Michael Bay	George Lucas
George Lucas	Alfred Hitchcock

Must alias table to be able to perform self-join?

Types of Joins



SQL JOINS

```
SELECT <select_list>
FROM TableA A
LEFT 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
RIGHT 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
```

© C.L. Moffatt, 2008

What’s the difference between a JOIN and an INNER JOIN?

How much information does a FULL OUTER JOIN provide?

QUESTIONS?