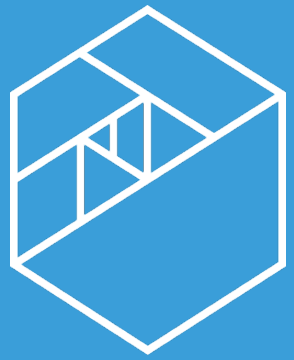


METIS

SQL FOUNDATIONS

JOINS PART I: CARTESIAN PRODUCT

METIS

Joining Data Tables

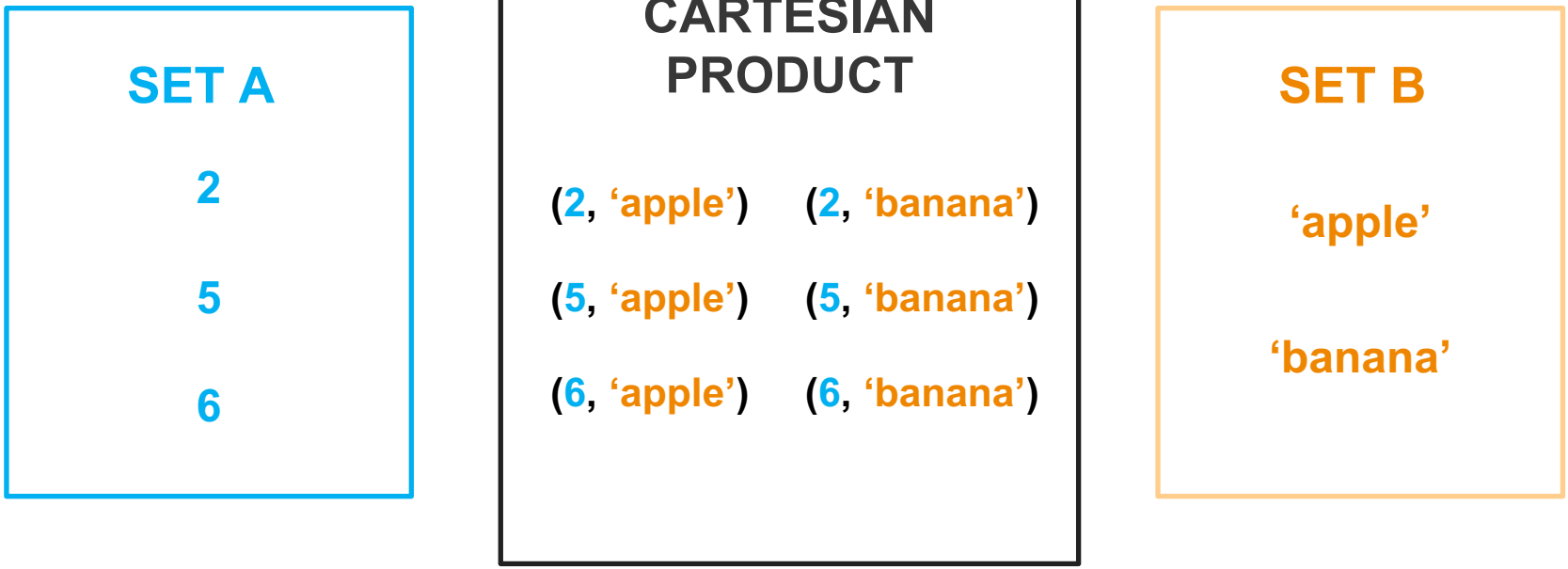
Question:
How much markup does the company charge for each item in stock?

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

Cartesian Product

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



Cartesian Product 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

Cartesian Product Example

Cartesian Product: Books x Directors

```
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.

Cartesian Product Filtered (Implicit Join)

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 PART 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      ← First table
    JOIN directors       ← Second table
     ON book_id = preferred_book_id;
```

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 (using *inventory* and *item_details*)

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	

Join these tables on ID with explicit inner join

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

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)

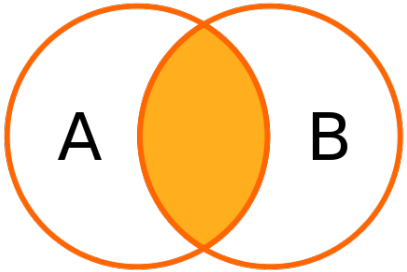
LEFT OUTER JOIN

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

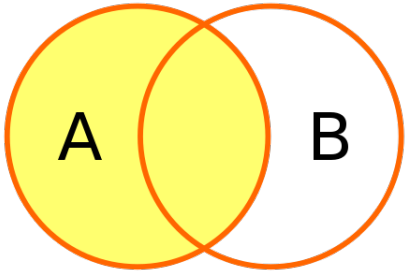
id	name	vendor_unit_price	item_id	name	sales_price
1	tiger t-shirt	4.2500	1	tiger t-shirt	9.9900
2	giraffe-print bag	24.9900	2	giraffe-print bag	49.9900
3	elephant tie	13.1900	3	elephant tie	35.4900
4	zebra-striped pants	16.8800	4	zebra-striped pants	30.9900
5	peacock feather hat	NULL	5	peacock feather 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

JOINS Graphically

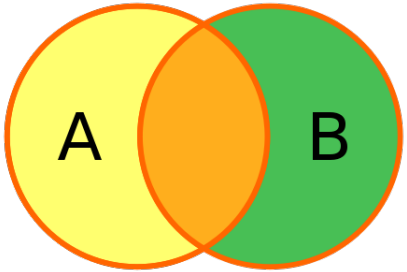
https://en.wikipedia.org/wiki/Join_(SQL)



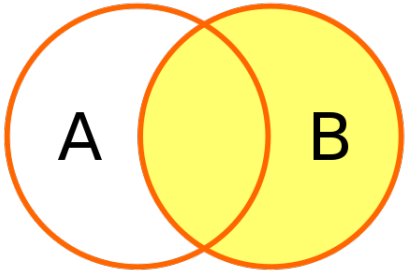
Inner Join



Left Outer Join



Full Outer Join



Right Outer Join
(less common)

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

EXERCISES: JOINS

METIS

Exercise: Joins

- The following questions concern the tables you created to store information on the activities offered by Company XYZ (first six rows shown below) and the vendors table for Company XYZ (next slide).

id	item	activity_level	category	family_friendly
1	wind surfing	4	sport	0
2	walk on Great Wall of China	2	site seeing	1
3	climb Mount Everest	5	sport	0
4	French cuisine package	0	food and beverage	1
5	geocaching package	1	sport	1
6	Broadway musical experience	0	culture	1

Note: Your table should have at least 10 rows (part of earlier exercise)

Exercise: Joins

- The table you created for Company XYZ's vendors.

id	vendor_name	phone_number	city	activity_id	price
1	Phil's Surfing Emporium	800-345-SURF	Honolulu	1	250.0000
2	Fun, Sun, and Surfing	888-541-1219	San Diego	1	300.0000
3	Trekking Everest	800-212-1001	Portland	3	14500.0000
4	Le Meilleur de la Mer	212-905-5521	New York	4	275.0000
5	Jacques et Lise	415-555-1000	San Francisco	4	199.9900
6	Live Ticket New York	347-333-SHOW	New York	6	350.0000

Exercise: Joins

Form the Cartesian product between the activities table and the vendors table for Company XYZ.

a. Select the top 10 rows from the Cartesian product (CP).

b. Does this CP seem useful to you? Why or why not.

c. Filter the CP to match up the activity ids. Is your result an example of an explicit or implicit inner join? Note: You will need to alias your tables and use the table.column convention in your filter since both tables have an "id" column.

Exercise: Joins

Perform an explicit inner join using the activities and vendors table by joining on activity id.

- Display the results as activity name, activity category, vendor name, and vendor price.
- Sort this joined table by vendor price to determine which vendor offers the most expensive item. What activity is it? What category of activity is it?
- Can you build a potential vendor list for each activity like the one below? Hint: You will need to use two group by statements. (Note: Your final result may look different than the example below depending on the additional vendors you included in your table.)

id	name	vendor_list
1	wind surfing	Phil's Surfing Emporium; Fun, Sun, and Surfing
3	climb Mount Everest	Trekking Everest
4	French cuisine package	Le Meilleur de la Mer; Jacques et Lise
6	Broadway musical experience	Live Ticket New York

Exercise: Joins

Join the activities and vendors tables on activity id in the following ways. Before executing each command, try to predict which rows will and will not be included. Where do you expect NULL values?

a. FULL OUTER JOIN

b. LEFT OUTER JOIN

c. RIGHT OUTER JOIN

SUBQUERIES

METIS

Subqueries

Item Details Table

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

Subqueries

Question:
What accessories sell for less than the elephant tie?

Step 1: What is the price of the elephant tie?

```
SELECT sales_price FROM item_details
    WHERE name LIKE '%elephant%';
```

sales_price
35.4900

Subqueries

Question:
What accessories sell for less than the elephant tie?

Step 2: What accessories sell for less than $35.49?

```
SELECT name, sales_price FROM item_details
    WHERE department = 'accessories'
    AND sales_price < 35.49;
```

name	sales_price
peacock feather hat	34.9900
leopard-print scarf	14.4900
aardvark earrings	9.9900

Subqueries with WHERE

Question:
What accessories sell for less than the elephant tie?

```
SELECT name, sales_price FROM item_details
    WHERE department = 'accessories'
    AND sales_price <
        (SELECT sales_price FROM item_details
         WHERE name LIKE '%elephant%');
```

Subquery: Must be enclosed in parentheses

name	sales_price
peacock feather hat	34.9900
leopard-print scarf	14.4900
aardvark earrings	9.9900

Subqueries with SELECT

Question:
What is the difference in price between all items and the least expensive item in the accessories department?

```
SELECT name, sales_price,
    sales_price - (SELECT min(sales_price)
FROM item_details WHERE department =
'accessories')
FROM item_details;
```

Subquery: Must be enclosed in parentheses

EXERCISES: SUBQUERIES

METIS

Exercise: Subqueries

Now that you are familiar with the tables for Company XYZ (activities and vendors) use information from them to answer the following questions. Be sure to use a subquery to answer each question in this section.

1. Several customers recently tried the wind surfing package but found it to be too action-packed. We'd like to suggest some other activities that have a lower activity level to them. Using a subquery to first select for the activity level of wind surfing, which items does the company offer with a lower activity rating?

2. What is the average activity level for all items offered by the company? (Check: Is your average really an integer?) Display the names and activity levels of items with levels that are less than average.

3. Which vendors offer packages for activities that are among the top 3 highest ranked in terms of activity level?

QUESTIONS?