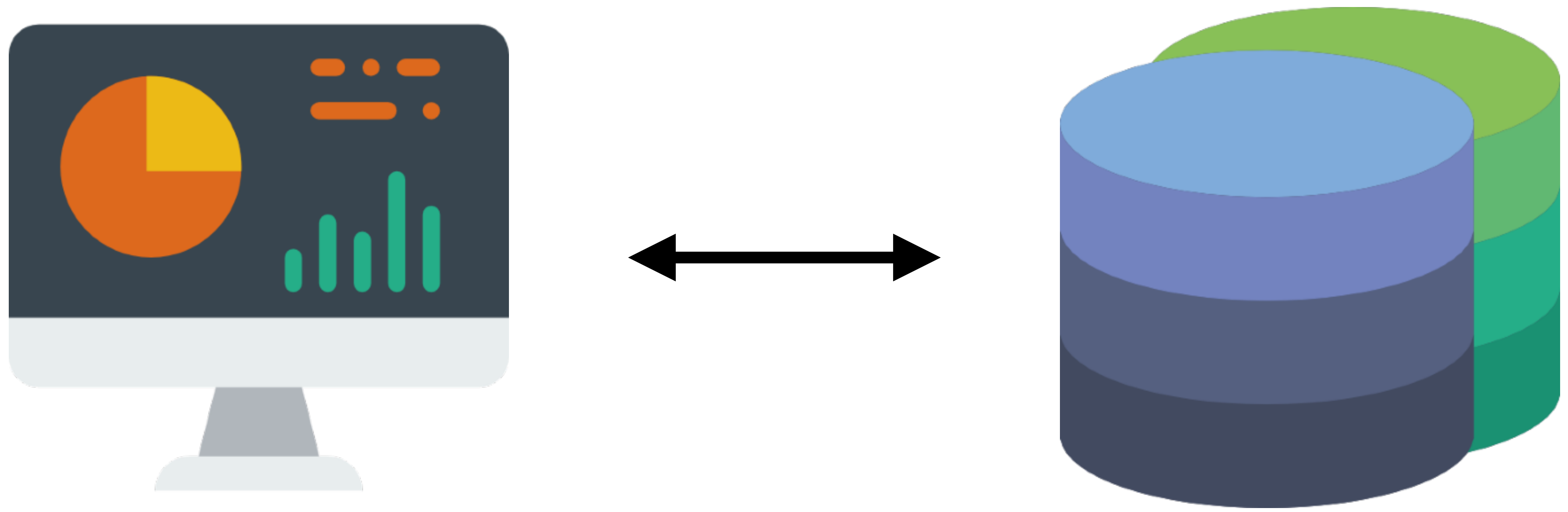
Advanced SQL

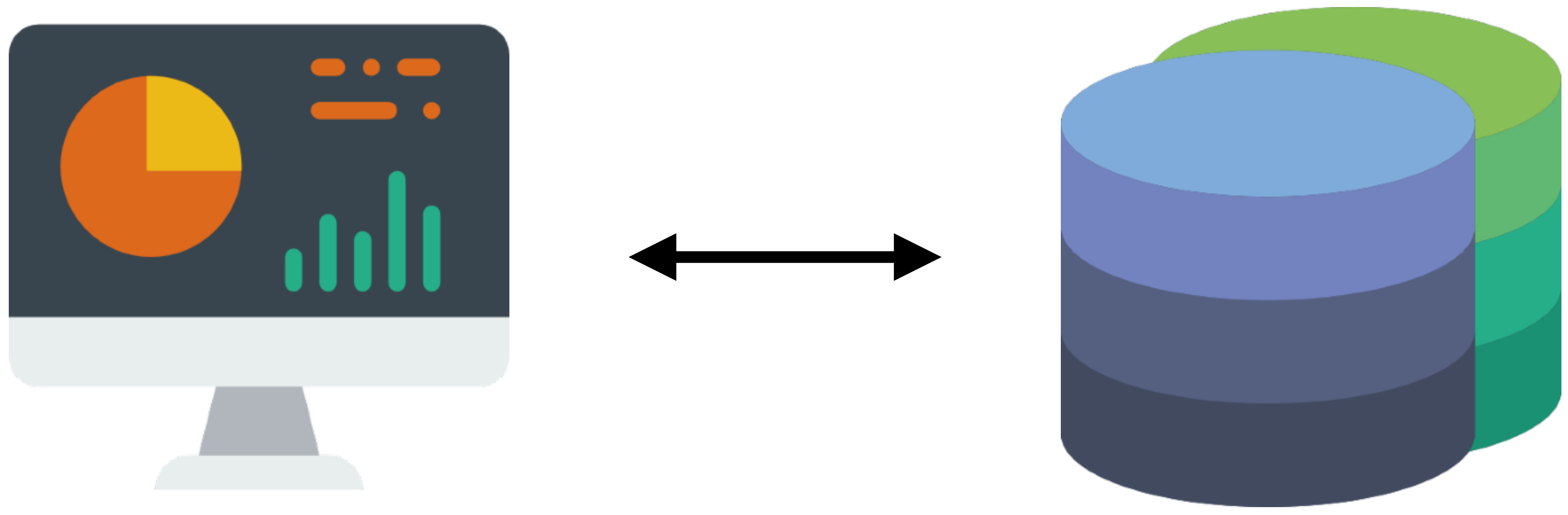
ctsai@DataLAB
Software Studio
2017 Spring

Why using DBMS?



[source]

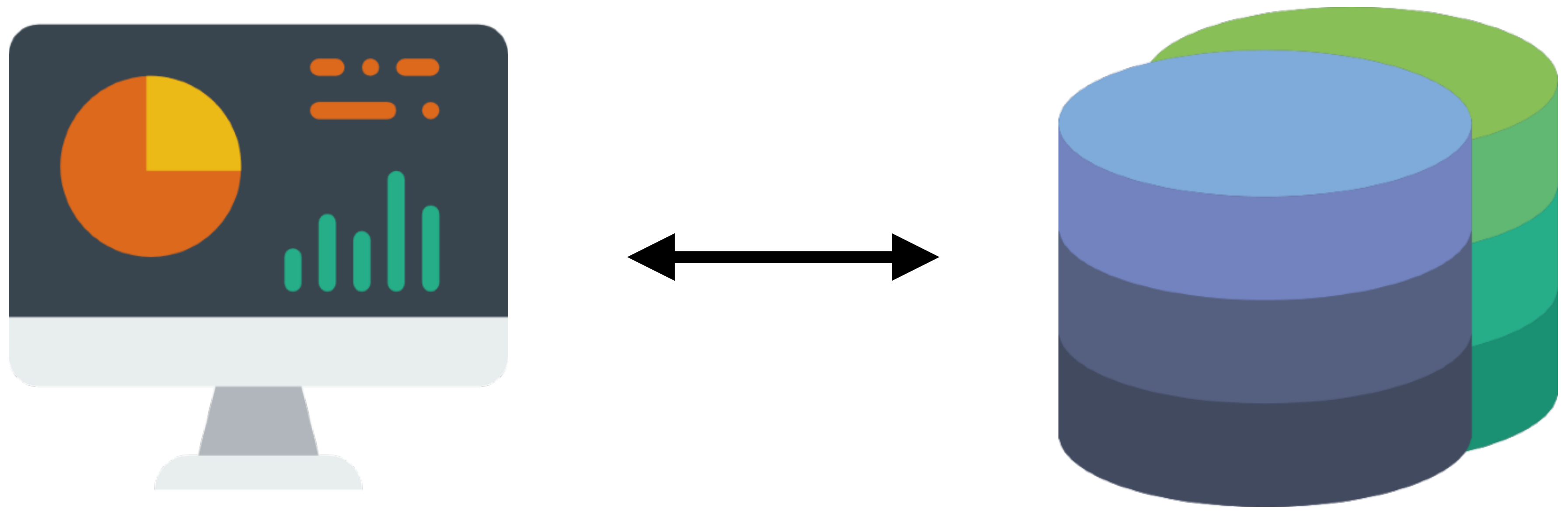
Why using DBMS?



From the client's point of view?

[source]

Why using DBMS?



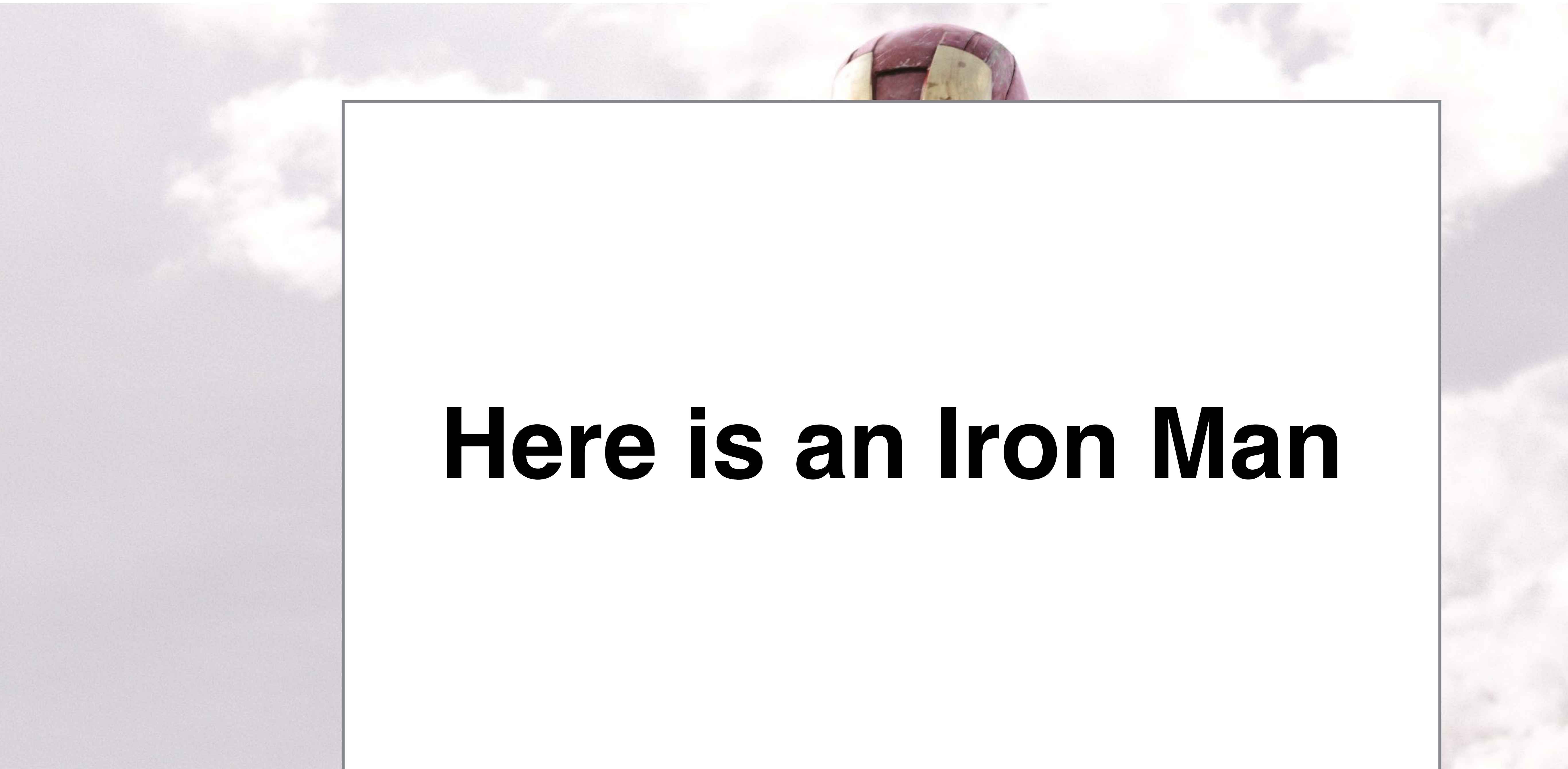
From the client's point of view?
From the developer's point of view?

[source]

Using DB wisely
Saves plenty of time

- Database are written by some of biggest company in the world

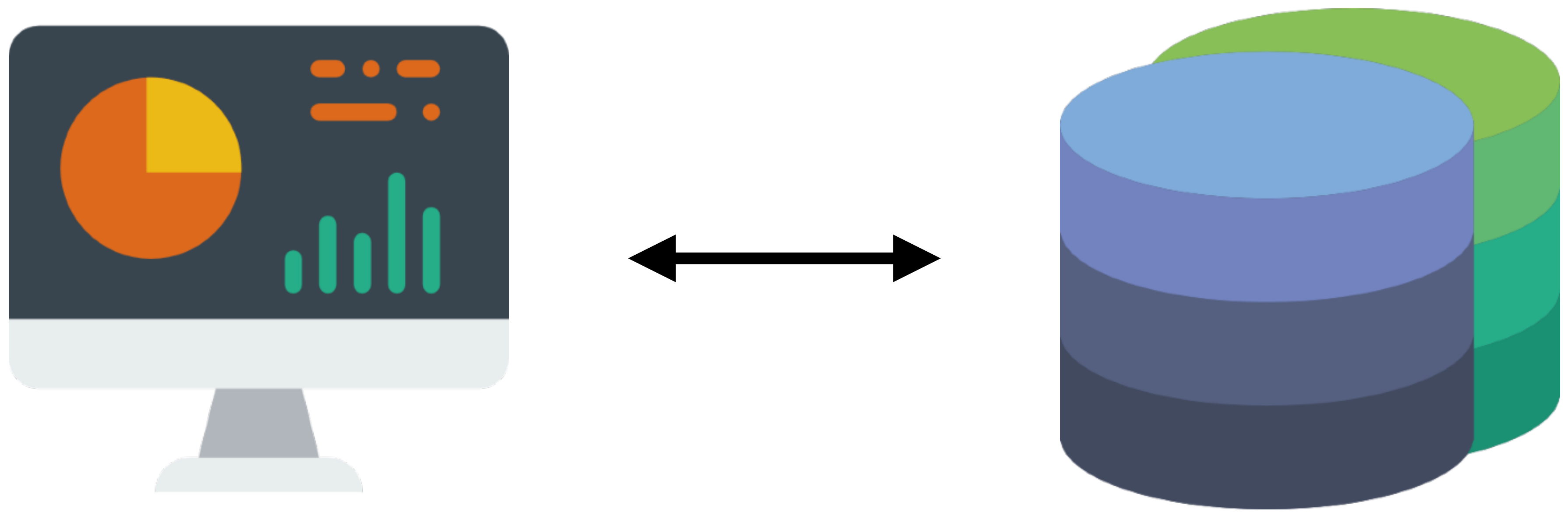
Using DB wisely saves you plenty of time



Here is an Iron Man

SQL

- To communicate to all database in the world, we need a standard language



[source]

Select Review

Student	
s_id	Primary key
s_name	名稱
s_level	等級
s_class	職業
s_lif	生命
s_atk	攻擊
s_def	防禦
s_mag	魔力
s_bs	伴侶

Select Review

Student	
s_id	Primary key
s_name	名稱
s_level	等級
s_class	職業
s_lif	生命
s_atk	攻擊
s_def	防禦
s_mag	魔力
s_bs	伴侶

What is primary key?

Select Review

Student	
s_id	Primary key
s_name	名稱
s_level	等級
s_class	職業
s_lif	生命
s_atk	攻擊
s_def	防禦
s_mag	魔力
s_bs	伴侶

- Which students' level more than 10?

```
SELECT * FROM student
WHERE s_level > 10
```

Select Review

Student	
s_id	Primary key
s_name	名稱
s_class	職業
s_level	等級
s_lif	生命
s_atk	攻擊
s_def	防禦
s_mag	魔力
s_bs	伴侶

Class	
c_id	Primary key
s_name	名稱
c_b_lif	生命加成
c_b_atk	攻擊加成
c_b_def	防禦加成
c_b_mag	魔力加成

Select Review

Student	
s_id	Primary key
s_name	名稱
s_level	等級
s_class	職業
s_b_lif	生命加成
s_b_atk	攻擊加成
s_b_def	防禦加成
s_b_mag	魔力加成
s_lif	生命
s_atk	攻擊
s_def	防禦
s_mag	魔力
s_bs	伴侶

Why is this schema design bad?

Query on multiple table

- Scenario :

  How to query a student’s information and class name at the same time?

```
SELECT * FROM student, class
WHERE s_id = 10
AND s_class = c_id;
```

Query on multiple table

- Scenario :

  How to query a student's information and class name at the same time?

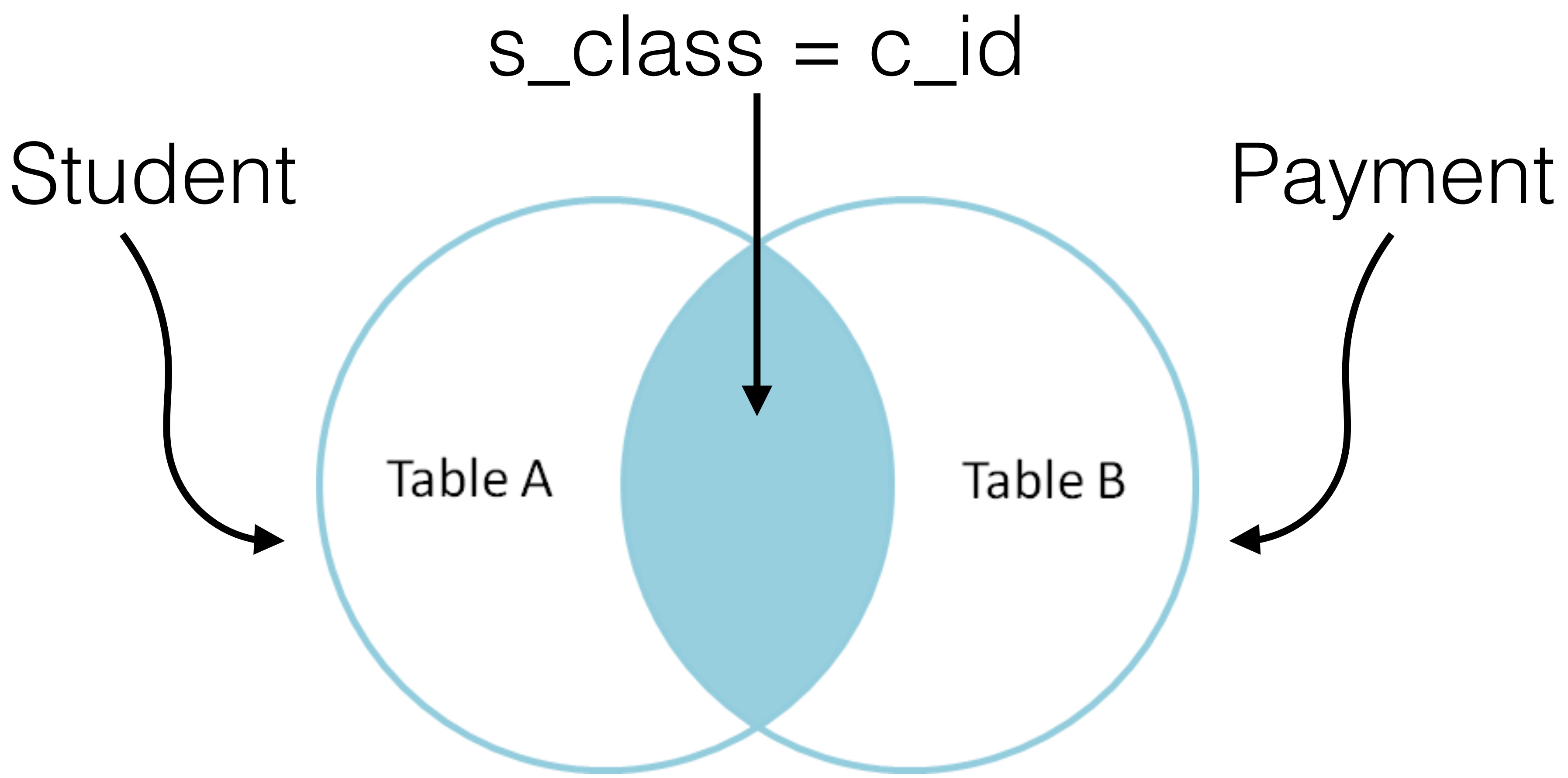
```
SELECT * FROM student, class
WHERE s_id = 10
AND s_class = c_id;
```

OR

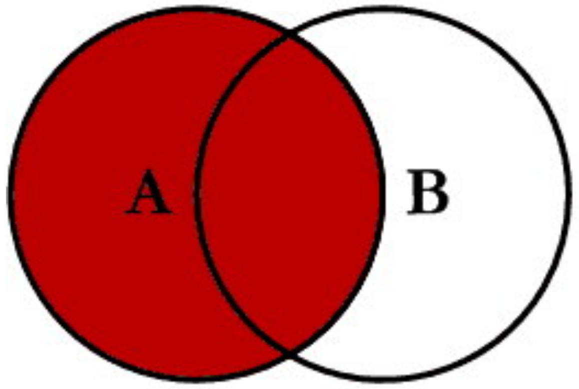
```
SELECT * FROM student
JOIN class ON s_class = c_id
WHERE s_id = 10 ;
```

Join

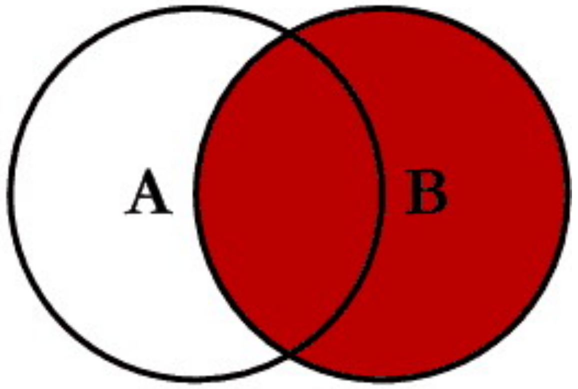
```
SELECT * FROM student
JOIN class ON s_class = c_id
WHERE s_id = 10 ;
```



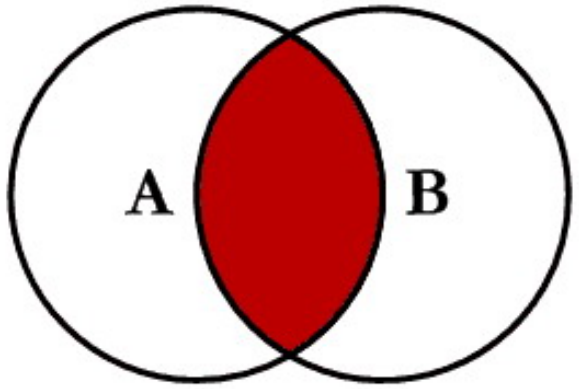
SQL JOINS



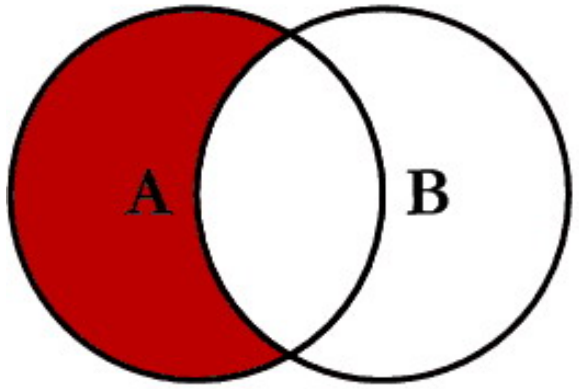
```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
```



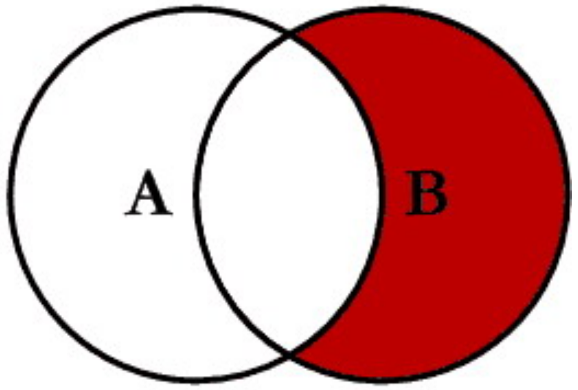
```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
```



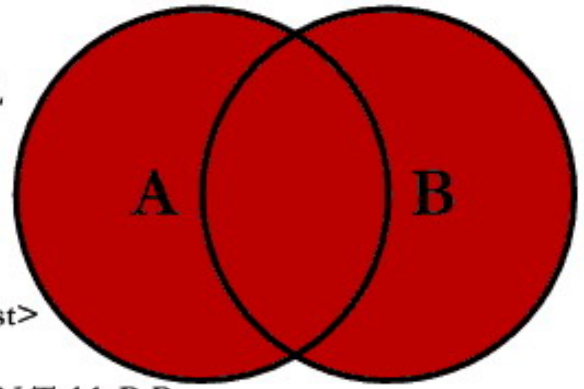
```
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
```



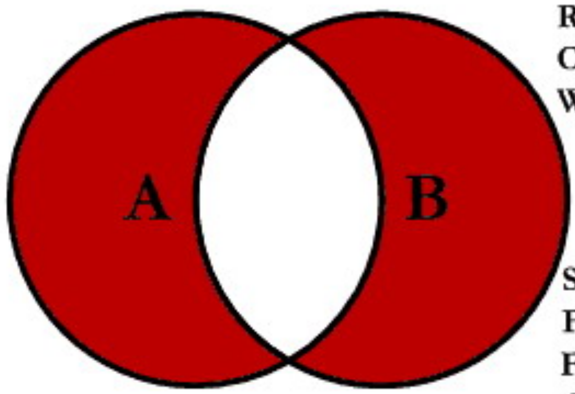
```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL
```



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

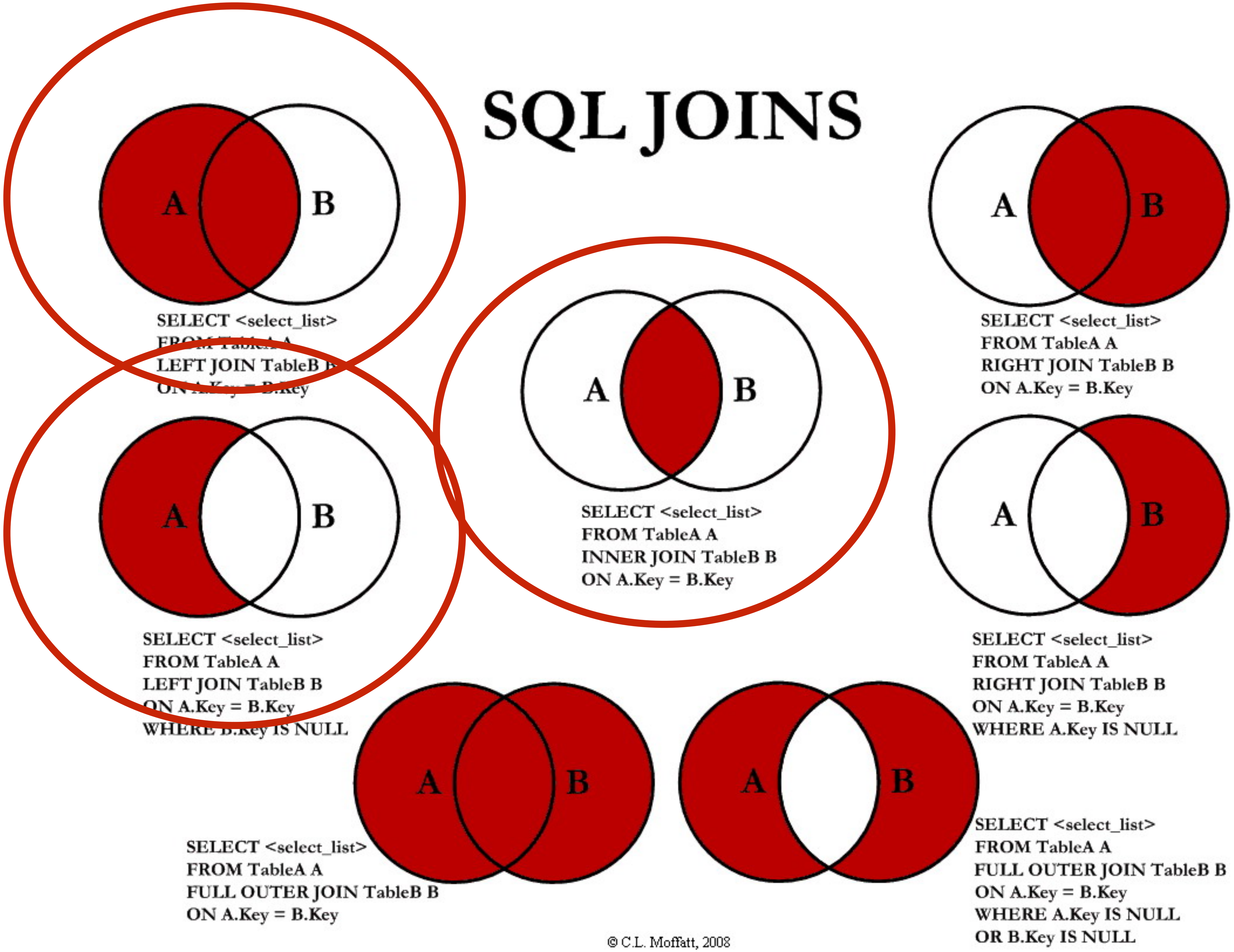


```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
```



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

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
```

Inner join

- Scenario :

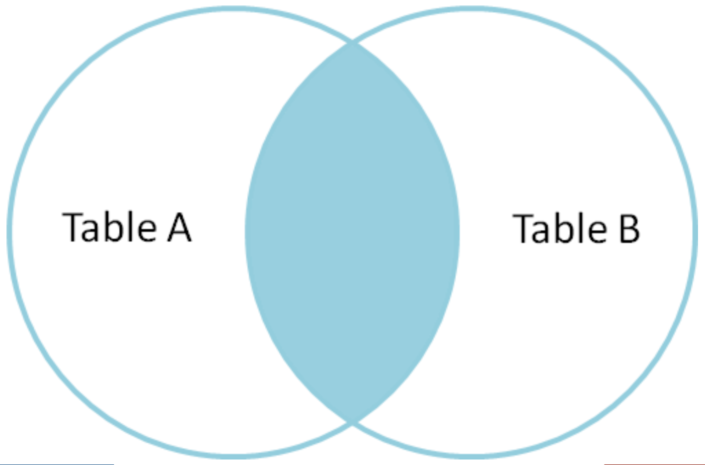
  How to query a payment with its buyer names?

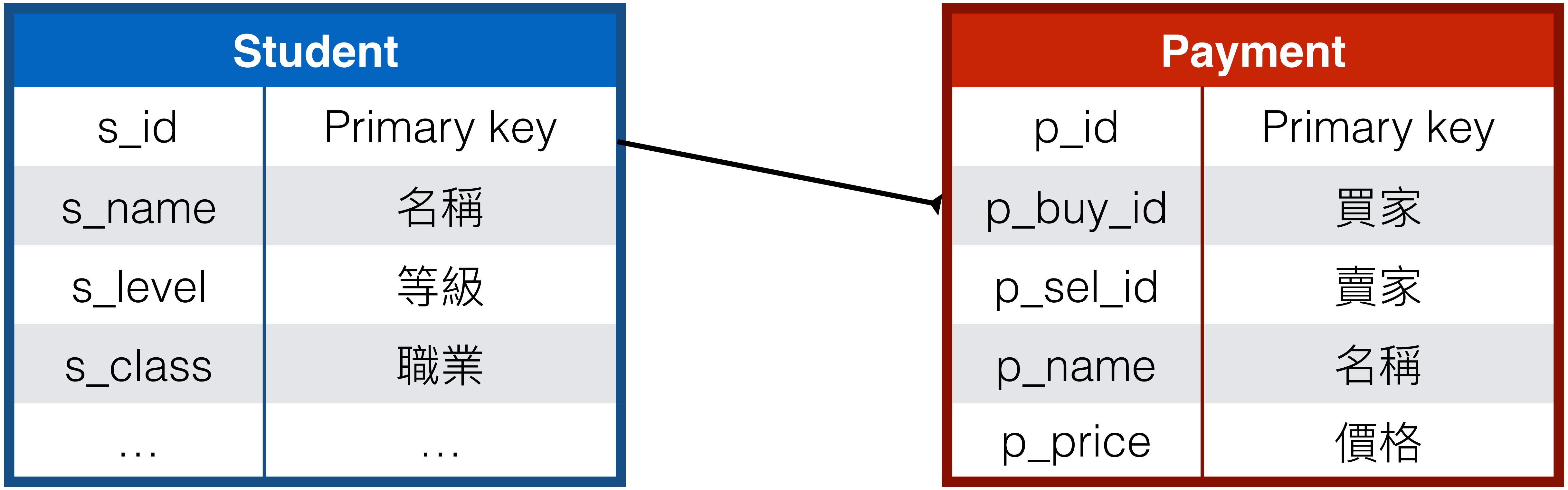
Payment	
p_id	Primary key
p_buy_id	買家
p_sel_id	賣家
p_name	名稱
p_price	價格

Inner join

- Scenario :

  How to query a payment with its **buyer names**?





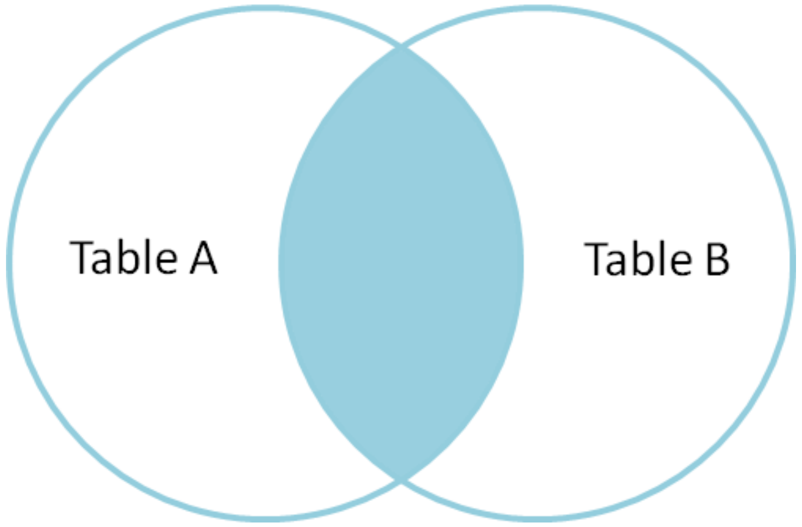
Student	
s_id	Primary key
s_name	名稱
s_level	等級
s_class	職業
…	…

Payment	
p_id	Primary key
p_buy_id	買家
p_sel_id	賣家
p_name	名稱
p_price	價格

Inner join

- Scenario :

  How to query a payment with its **buyer names**?

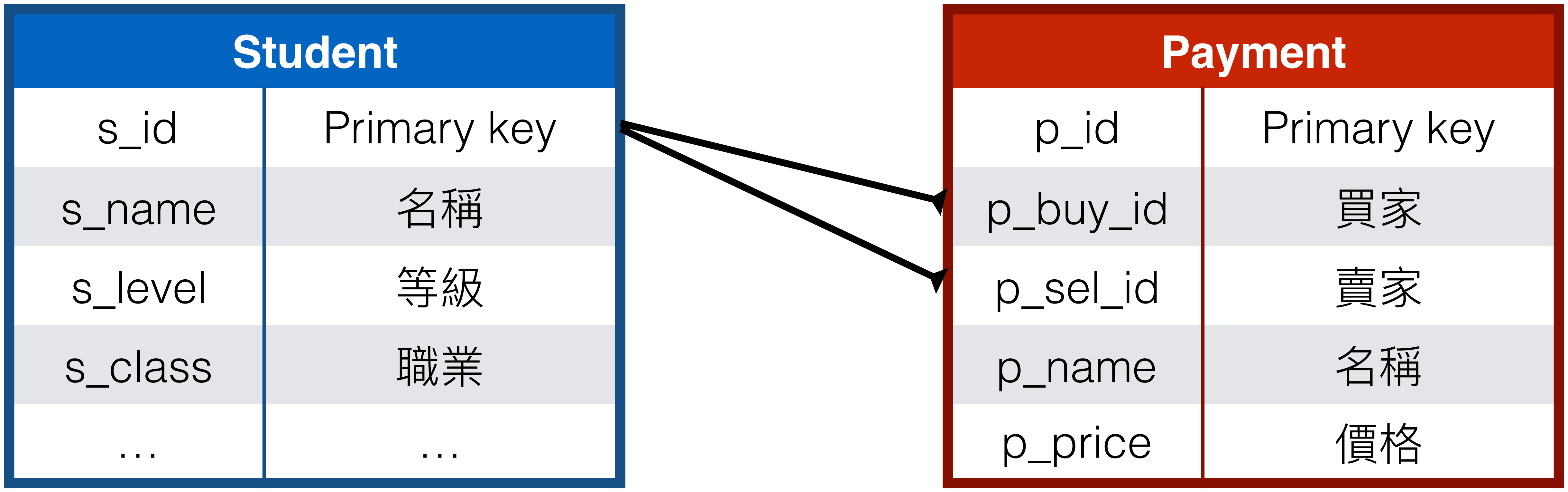


```
SELECT s_name, p_name FROM student
INNER JOIN payment on s_id = p_buy_id;
```

Inner join

- Scenario :

  How to query a payment with its **buyer names** and **seller names**?



Student	
s_id	Primary key
s_name	名稱
s_level	等級
s_class	職業
…	…

Payment	
p_id	Primary key
p_buy_id	買家
p_sel_id	賣家
p_name	名稱
p_price	價格

Inner join

- Scenario :

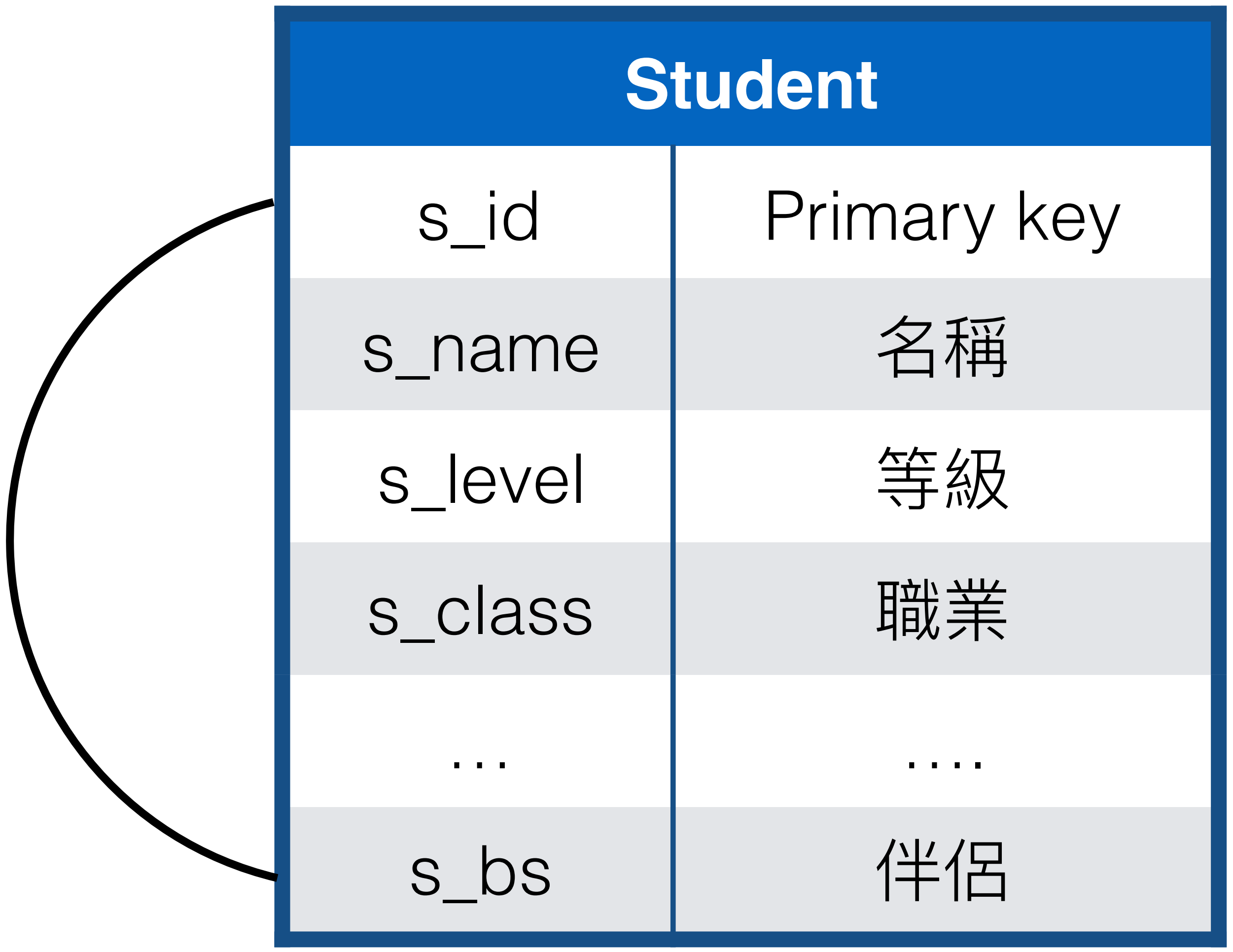
  How to query a payment with its **buyer names** and **seller names**?

```
SELECT s1.s_name buyer, p_name
       , s2.s_name seller
FROM student s1 INNER JOIN payment
on s1.s_id = p_buy_id
INNER JOIN student s2
on s2.s_id = p_sel_id;
```

Self Join

- Scenario :

  How to get best friends pairs in student?



Student	
s_id	Primary key
s_name	名稱
s_level	等級
s_class	職業
…	….
s_bs	伴侶

Self Join

- Scenario :

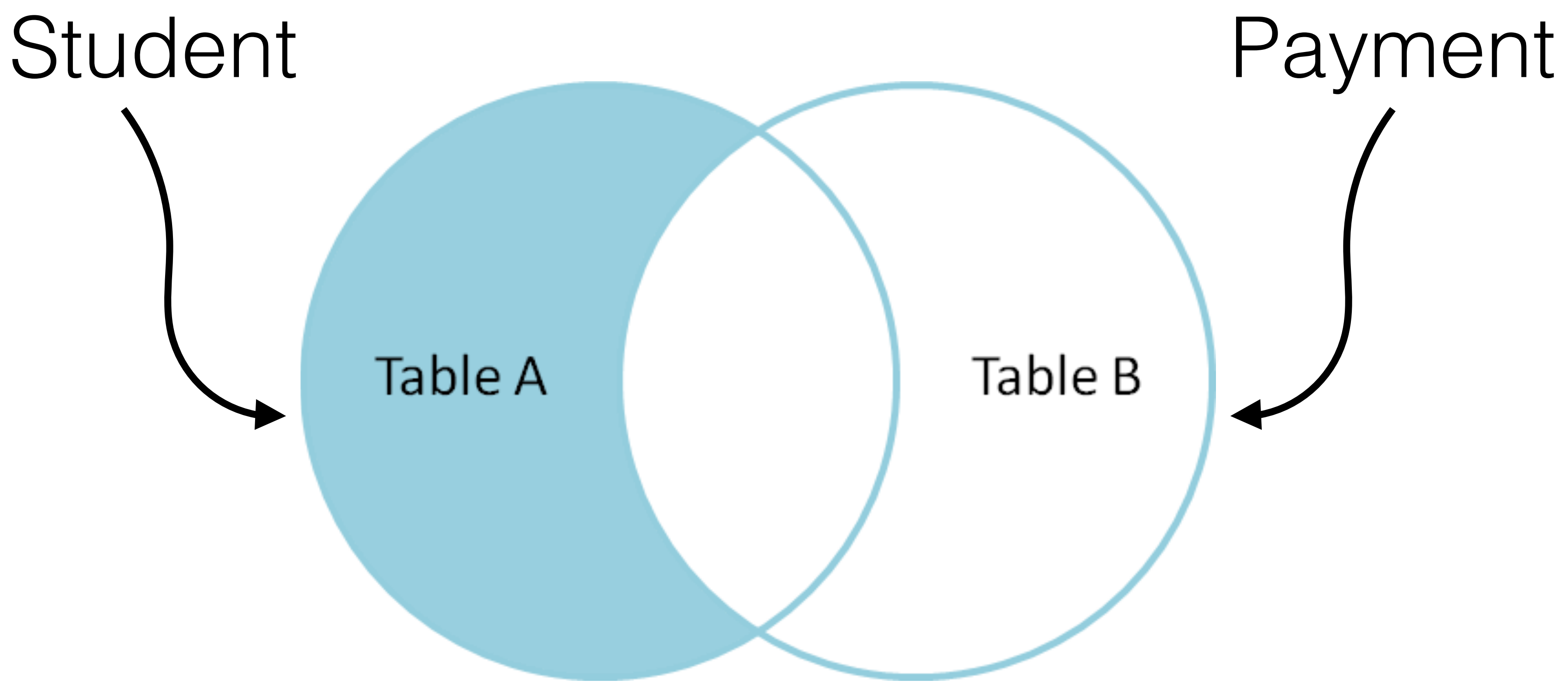
  How to get best friends pairs in student?
- Same as the previous join

```
SELECT s1.s_name, s2.s_name
FROM student s1
INNER JOIN student s2
ON s1.s_bs = s2.s_id;
```

Left outer join

- Scenario :

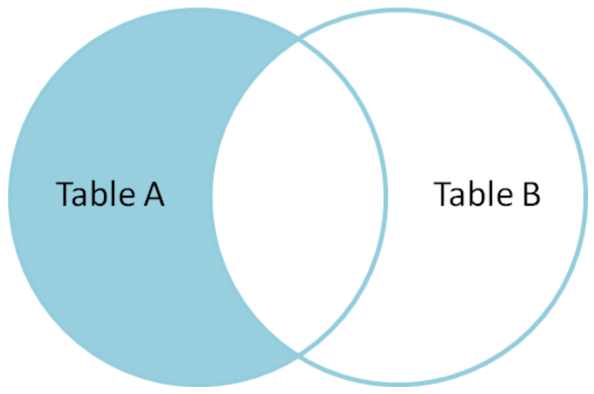
  Who haven't buy an item?



Left outer join

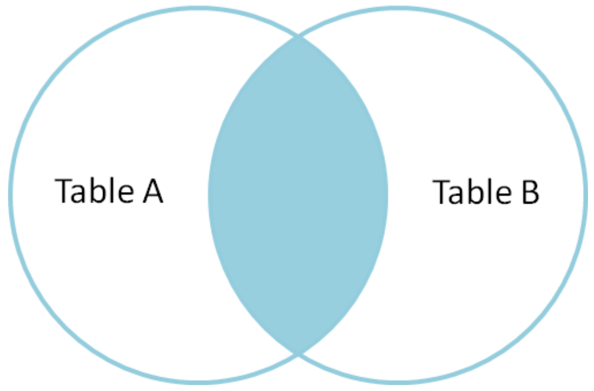
- Unfortunately, SQL don’t have native left outer join
- But SQL have left join !

Left-Outer Join

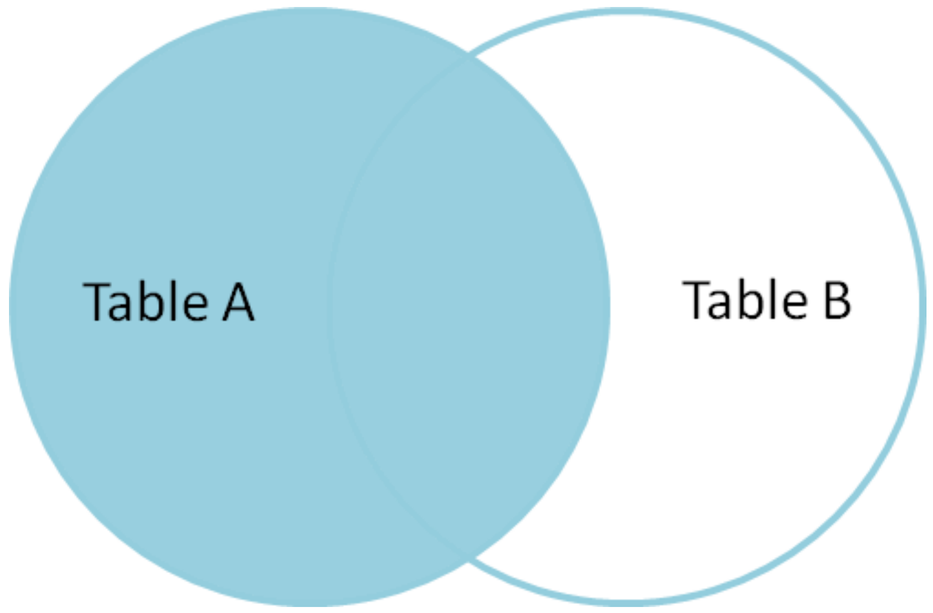


+

Inner Join



=



Left Join

Left outer join

- Scenario :

  How to query a payment with its buyer names?

```
SELECT * FROM student
LEFT JOIN payment on s_id = p_buy_id
WHERE payment.p_buy_id is NULL;
```

Only select students that don’t have NULL p_buy_id

Left outer join

- Scenario :

  How to query a payment with its buyer names?

Left Join

```
SELECT * FROM student
LEFT JOIN payment on s_id = p_buy_id
WHERE payment.p_buy_id is NULL;
```

Only select students that don’t have NULL p_buy_id

Left outer join

- Scenario :

  How to query a payment with its buyer names?

Left Outer Join

Left Join

```
SELECT * FROM student
LEFT JOIN payment on s_id = p_buy_id
WHERE payment.p_buy_id is NULL;
```

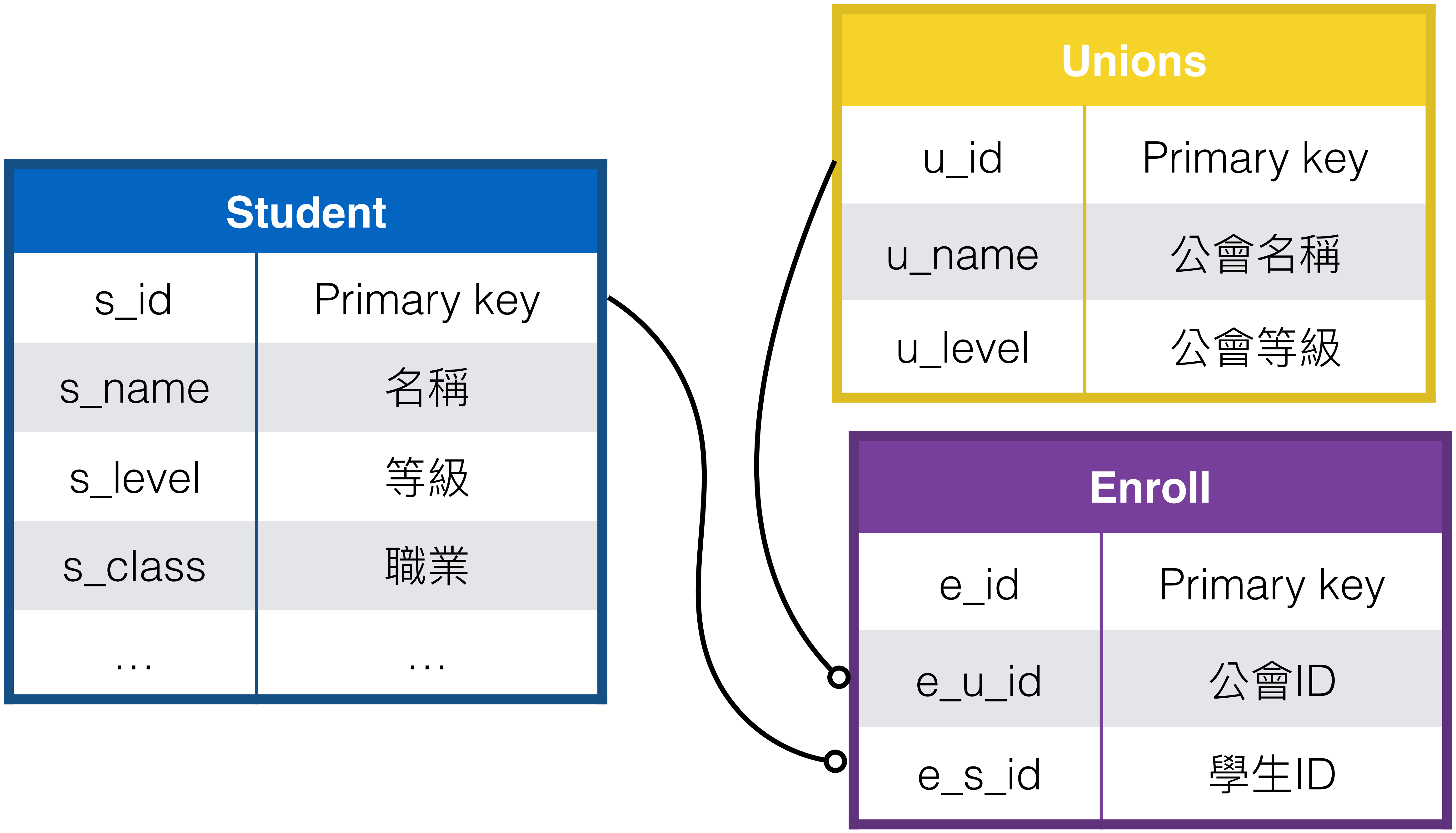
Only select students that don't have NULL p_buy_id

Why not store multiple key in one field ?

Student	
s_id	Primary key
s_name	名稱
s_level	等級
s_class	職業
…	….
s_unions	1,2…

Unions	
u_id	Primary key
u_name	公會名稱
u_level	公會等級

Why not store multiple key in one field ?



Student

s_id	Primary key
s_name	名稱
s_level	等級
s_class	職業
…	…

Unions

u_id	Primary key
u_name	公會名稱
u_level	公會等級

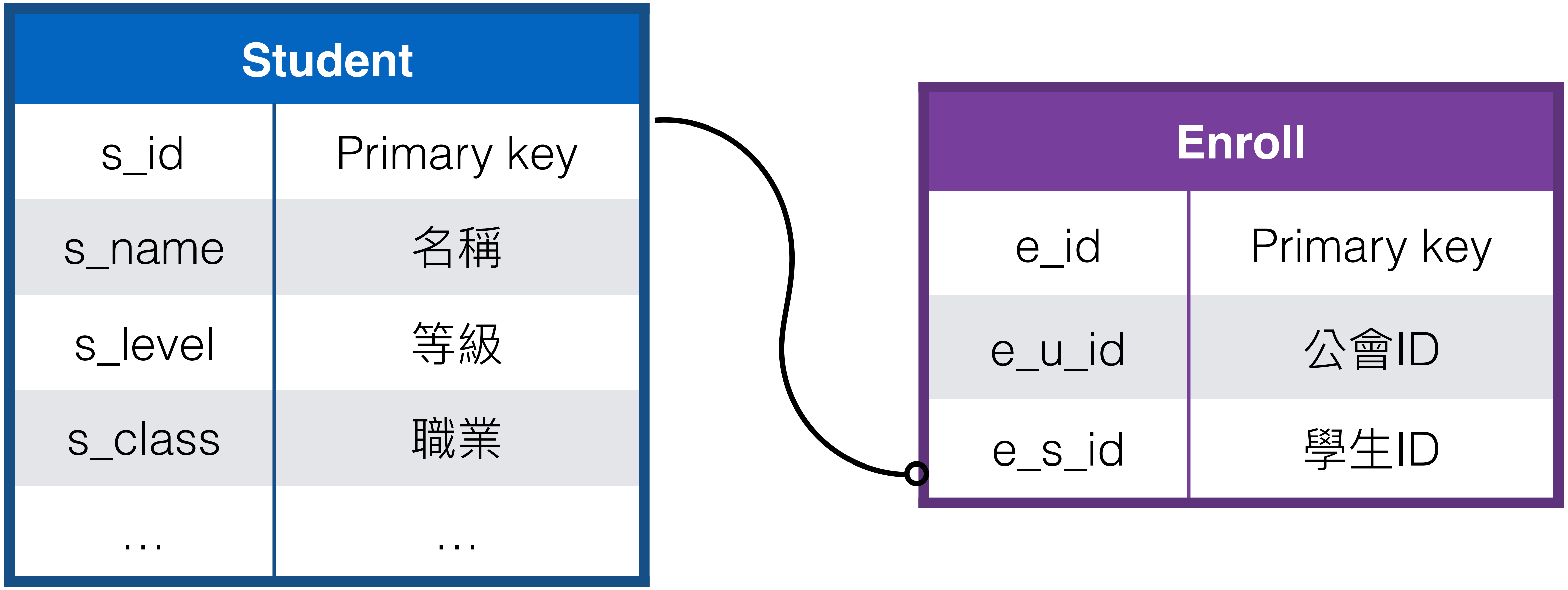
Enroll

e_id	Primary key
e_u_id	公會ID
e_s_id	學生ID

Group By and Aggregation

- Scenario :

  What is sum of attack in a union?



Student	
s_id	Primary key
s_name	名稱
s_level	等級
s_class	職業
…	…

Enroll	
e_id	Primary key
e_u_id	公會ID
e_s_id	學生ID

Group By and Aggregation

- Scenario :

  What is sum of attack in a union?

```
SELECT e_u_id, sum(s_atk) FROM student
INNER JOIN enroll on s_id = e_s_id
GROUP BY e_u_id;
```

Enroll	
e_id	Primary key
e_u_id	公會ID
e_s_id	學生ID

Having ? Where?

Pre-Filter

WHERE

<Condition on field>

$\downarrow$

Aggregation Function

SUM, COUNT

$\downarrow$

Post-Filter

HAVING

<Condition on aggregation result>

Data Flow $\downarrow$

Having ? Where?

- Scenario :

  Which unions that sum of attack more than 300?

```
SELECT e_u_id , sum(s_atk) FROM student
INNER JOIN enroll on s_id = e_s_id
GROUP BY e_u_id HAVING sum(s_atk) > 300;
```

  Which is the sum of life of the 打醬油 in a unions?

```
SELECT e_u_id , sum(s_lif) FROM student
INNER JOIN enroll on s_id = e_s_id
WHERE s_class = 3
GROUP BY e_u_id;
```