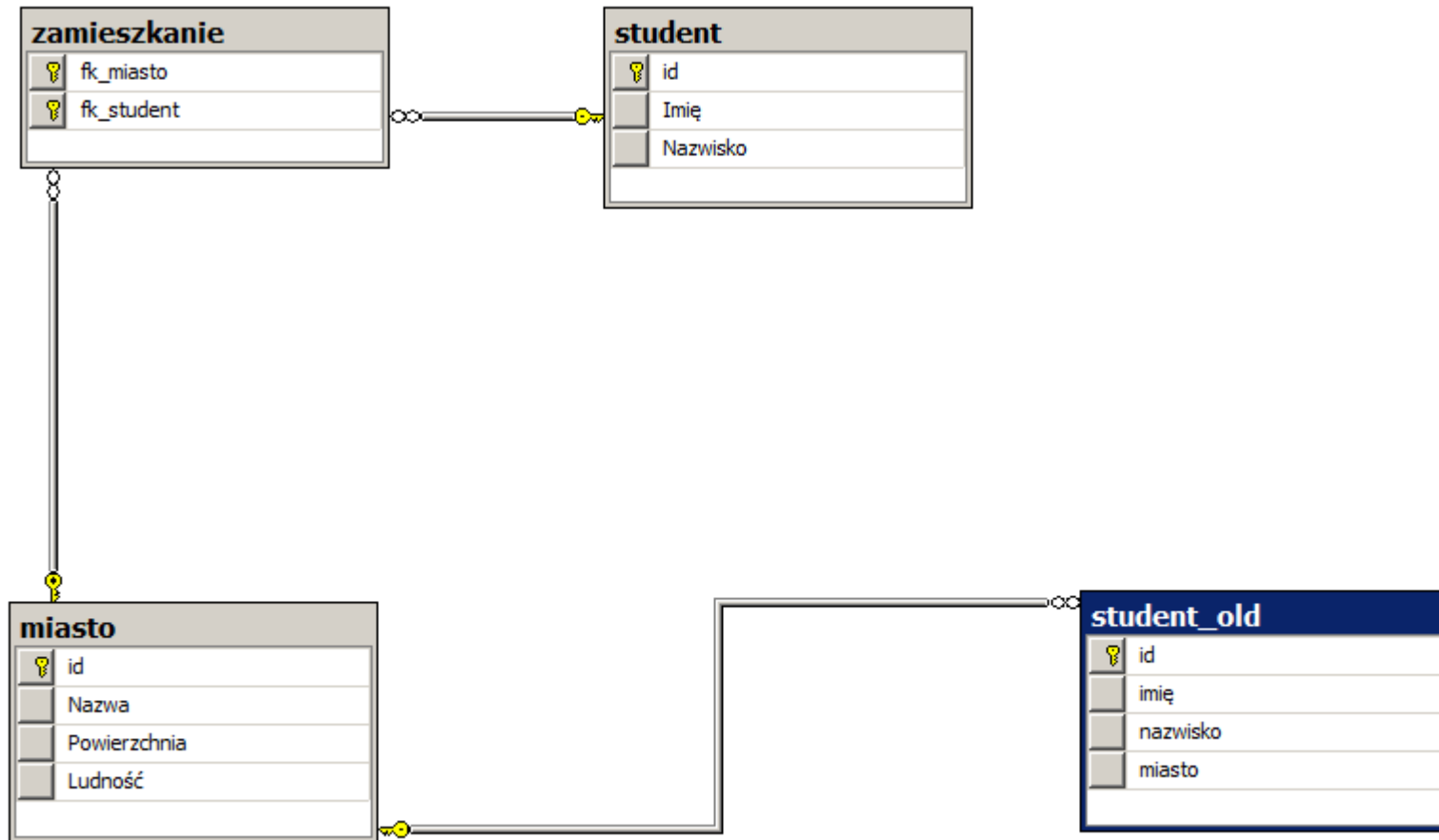


# Bazy danych

Dr inż. Michał Kruk

# Przykładowa baza



# Złączenia

- Jedną z podstawowych i najczęściej wykorzystywanych cech języka SQL jest możliwość łączenia tabel.

# Przykładowy diagram

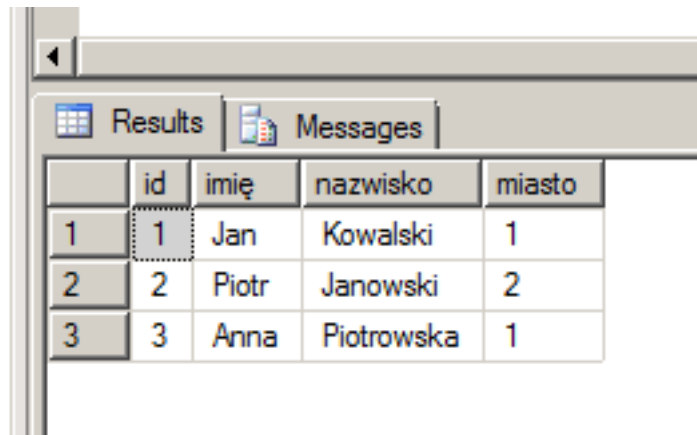


# Problem ze zwykłym selectem

- Użycie „zwykłego” selecta:

**Select \* from student**

Zwróci informacje o studencie, ale otrzymamy dodatkowo tylko id miasta:



	id	imię	nazwisko	miasto
1	1	Jan	Kowalski	1
2	2	Piotr	Janowski	2
3	3	Anna	Piotrowska	1

# Problem ze zwykłym selectem

- Często istnieje potrzeba wyświetlenia obydwu informacji – o studencie i obok o mieście w którym mieszka
- Do tego celu należy złączyć obydwie tabele
- Łączenie wewnętrzne tabel (INNER JOIN) - jeżeli nie zostało to inaczej sprecyzowane, to domyślnie łączenie tabel w zapytaniu SQL jest traktowane jako łączenie wewnętrzne (INNER JOIN lub po prostu JOIN) Łączenie wewnętrzne kojarzy ze sobą rekordy występujące w dwóch lub więcej tabelach i zwraca w wyniku tylko pasujące rekordy.

# INNER JOIN

Przykładowe zapytanie:

- **SELECT \* FROM tab1, tab2 WHERE tab1.ID = tab2.ID**
- **SELECT \* FROM tab1 INNER JOIN tab2 ON tab1.ID = tab2.ID**

# INNER JOIN

- Samo polecenie

**SELECT \* FROM student, miasto**

Złączy obydwie tabele, ale w wyniku otrzymamy iloczyn kartezjański – każdy wiersz z jednej tabeli z każdym wierszem drugiej tabeli



# INNER JOIN

Results					Messages			
	id	imię	nazwisko	miasto				
1	1	Jan	Kowalski	1				
2	2	Piotr	Janowski	2				
3	3	Anna	Piotrowska	1				

	id	Nazwa	Powierzchnia	Ludność
1	1	Warszawa	10	20000000
2	2	Kraków	5	1000000

	id	imię	nazwisko	miasto	id	Nazwa	Powierzchnia	Ludność
1	1	Jan	Kowalski	1	1	Warszawa	10	20000000
2	2	Piotr	Janowski	2	1	Warszawa	10	20000000
3	3	Anna	Piotrowska	1	1	Warszawa	10	20000000
4	1	Jan	Kowalski	1	2	Kraków	5	1000000
5	2	Piotr	Janowski	2	2	Kraków	5	1000000
6	3	Anna	Piotrowska	1	2	Kraków	5	1000000

# INNER JOIN

- Dlatego niezbędne jest dodanie ograniczenia:

**SELECT \* FROM student, miasto where  
student.miasto = miasto.id**

lub ( to samo )

**select \* from student\_old inner join miasto on  
student\_old.miasto = miasto.id**

# INNER JOIN



The screenshot shows a database query results window with two tabs: 'Results' and 'Messages'. The 'Results' tab is active, displaying a table with 9 columns and 3 rows. The columns are: an index column (1, 2, 3), 'id', 'imię', 'nazwisko', 'miasto', 'id', 'Nazwa', 'Powierzchnia', and 'Ludność'. The data represents an inner join of two tables. The first table has columns 'id', 'imię', 'nazwisko', and 'miasto'. The second table has columns 'id', 'Nazwa', 'Powierzchnia', and 'Ludność'. The join is performed on the 'id' columns. The first row shows a match between id=1 in both tables, resulting in Jan Kowalski from Warszawa. The second row shows a match between id=2, resulting in Piotr Janowski from Kraków. The third row shows a match between id=1, resulting in Anna Piotrowska from Warszawa.

	id	imię	nazwisko	miasto	id	Nazwa	Powierzchnia	Ludność
1	1	Jan	Kowalski	1	1	Warszawa	10	20000000
2	2	Piotr	Janowski	2	2	Kraków	5	1000000
3	3	Anna	Piotrowska	1	1	Warszawa	10	20000000

# Aliasy

```
select * from student_old
```

```
inner join miasto
```

```
on student_old.miasto = miasto.id
```

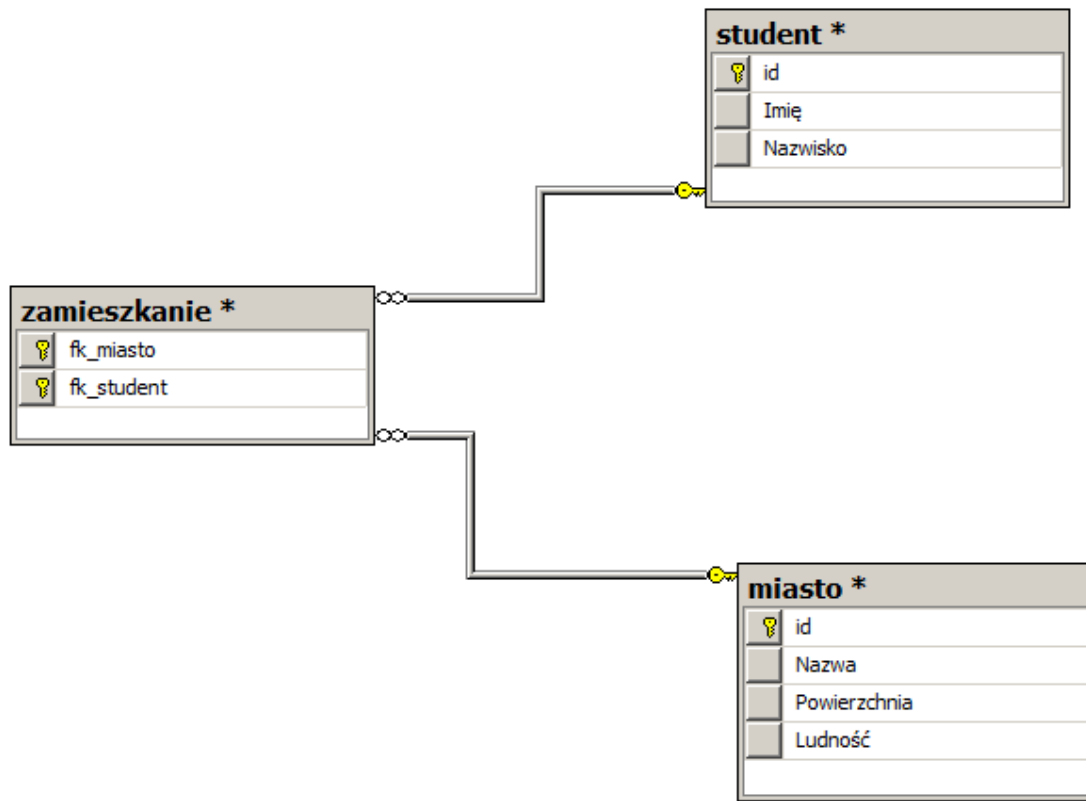
Znacznie wygodniej jest stosować aliasy:

```
select * from student_old as s
```

```
inner join miasto as m
```

```
on s.miasto = m.id
```

# INNER JOIN



# INNER JOIN

- W przypadku złączenia tabel będących w relacji wiele do wielu, a więc 3, złączenie obejmie 3 tabele:

**select \* from student**

**inner join zamieszkanie**

**on student.id=zamieszkanie.fk\_student**

**inner join miasto**

**on miasto.id=zamieszkanie.fk\_miasto**

# INNER JOIN

Results Messages									
	id	Imię	Nazwisko						
1	1	Jan	Kowalski						
2	2	Piotr	Janowski						
3	3	Anna	Piotrowska						

	id	Nazwa	Powierzchnia	Ludność
1	1	Warszawa	10	20000000
2	2	Kraków	5	1000000

	id	Imię	Nazwisko	fk_miasto	fk_student	id	Nazwa	Powierzchnia	Ludność
1	1	Jan	Kowalski	1	1	1	Warszawa	10	20000000
2	2	Piotr	Janowski	1	2	1	Warszawa	10	20000000
3	3	Anna	Piotrowska	1	3	1	Warszawa	10	20000000
4	1	Jan	Kowalski	2	1	2	Kraków	5	1000000

# Złączenie zewnętrzne - OUTER JOIN

- łączenie zewnętrzne tabel jest rozszerzeniem złączenia INNER JOIN.
- OUTER JOIN pozwala na włączenie do tabeli wynikowej rekordów które kwalifikują się do tabeli wynikowej na podstawie polecenia INNER JOIN i dodatkowo włączyć wybrane rekordy nie spełniające warunków zapytania.



# OUTER JOIN

- Złączenie zewnętrzne może przybrać następujące postacie:
- Złączenie lewe - LEFT OUTER JOIN W złączeniu zewnętrznym lewym zwracane są wszystkie wiersze występujące w tabeli z lewej strony, a wiersze z prawej tabeli, które nie zostały znalezione, wypełnione są wartościami null.

```
SELECT * FROM Tab1 LEFT OUTER JOIN Tab2 ON  
Tab1.ID = Tab2.ID
```

# OUTER JOIN

```
select * from student  
inner join zamieszkanie  
on student.id=zamieszkanie.fk_student  
left outer join miasto  
on miasto.id=zamieszkanie.fk_miasto
```

# OUTER JOIN

Results Messages									
	id	Imię	Nazwisko						
1	1	Jan	Kowalski						
2	2	Piotr	Janowski						
3	3	Anna	Piotrowska						

	id	Nazwa	Powierzchnia	Ludność
1	1	Warszawa	10	20000000
2	2	Kraków	5	1000000
3	3	Łódź	3	800000

	id	Imię	Nazwisko	fk_miasto	fk_student	id	Nazwa	Powierzchnia	Ludność
1	1	Jan	Kowalski	1	1	1	Warszawa	10	20000000
2	2	Piotr	Janowski	1	2	1	Warszawa	10	20000000
3	3	Anna	Piotrowska	1	3	1	Warszawa	10	20000000
4	1	Jan	Kowalski	2	1	2	Kraków	5	1000000

# OUTER JOIN

```
select * from student  
inner join zamieszkanie  
on student.id=zamieszkanie.fk_student  
right outer join miasto  
on miasto.id=zamieszkanie.fk_miasto
```

# OUTER JOIN

Results				Messages			
	id	Imię	Nazwisko				
1	1	Jan	Kowalski				
2	2	Piotr	Janowski				
3	3	Anna	Piotrowska				

	id	Nazwa	Powierzchnia	Ludność
1	1	Warszawa	10	20000000
2	2	Kraków	5	1000000
3	3	Łódź	3	800000

	id	Imię	Nazwisko	fk_miasto	fk_student	id	Nazwa	Powierzchnia	Ludność
1	1	Jan	Kowalski	1	1	1	Warszawa	10	20000000
2	2	Piotr	Janowski	1	2	1	Warszawa	10	20000000
3	3	Anna	Piotrowska	1	3	1	Warszawa	10	20000000
4	1	Jan	Kowalski	2	1	2	Kraków	5	1000000
5	NULL	NULL	NULL	NULL	NULL	3	Łódź	3	800000

# Złączenie prawe - RIGHT OUTER JOIN

- Złączenie prawe - RIGHT OUTER JOIN
- W złączeniu zewnętrznym prawym zwracane są wszystkie wiersze występujące w tabeli z prawej strony, a wiersze z lewej tabeli, które nie zostały znalezione, wypełnione są wartościami null.

```
SELECT * FROM Tab1 RIGHT OUTER JOIN Tab2  
ON Tab1.ID = Tab2.ID
```

# Złączenie pełne - FULL OUTER JOIN

- W rezultatach zapytania SQL obecne są wiersze połączone wewnętrznie, rozszerzone o rekordy nie zwrócone z obydwu tabel, wypełnione nullami

```
SELECT * FROM Tab1 FULL OUTER JOIN Tab2  
ON Tab1.ID = Tab2.ID
```

# FULL OUTER JOIN

```
select * from student  
full outer join zamieszkanie  
on student.id=zamieszkanie.fk_student  
full outer join miasto  
on miasto.id=zamieszkanie.fk_miasto
```



# FULL OUTER JOIN

Results Messages										
	id	Imię	Nazwisko							
1	1	Jan	Kowalski							
2	2	Piotr	Janowski							
3	3	Anna	Piotrowska							

	id	Nazwa	Powierzchnia	Ludność
1	1	Warszawa	10	20000000
2	2	Kraków	5	1000000
3	3	Łódź	3	800000

	id	Imię	Nazwisko	fk_miasto	fk_student	id	Nazwa	Powierzchnia	Ludność
1	1	Jan	Kowalski	1	1	1	Warszawa	10	20000000
2	1	Jan	Kowalski	2	1	2	Kraków	5	1000000
3	2	Piotr	Janowski	1	2	1	Warszawa	10	20000000
4	3	Anna	Piotrowska	1	3	1	Warszawa	10	20000000
5	NULL	NULL	NULL	NULL	NULL	3	Łódź	3	800000

# UNION

- Zdarzają się przypadki, że chcesz obejrzeć rezultaty serii zapytań połączone razem; użyj unii (**UNION**).
- Seria kilku zapytań połączona unią musi zwracać taką samą liczbę kolumn!

```
select imię,nazwisko from student
```

```
union
```

```
select nazwa,cast(ludność as varchar(50))  
from miasto
```

# UNION

- Jak pokazano w przykładzie w obydwu zapytaniach musi być zgodna liczba kolumn oraz typ zwracanych kolumn
- Użycie UNION automatycznie eliminuje występowanie duplikatów ( danych powtarzających się ) – nie trzeba stosować **distinct**