JOIN

(połączenia)

dimon.work/kurs.html

Połączenia

Jeśli między tabelami w bazie danych (BD) istnieją połączenia, czyli relacje, to w pewnym momencie możemy potrzebować uzyskać dane nie tylko z jednej konkretnej tabeli, ale jednocześnie z kilku tabel. W takim przypadku używa się operacji **JOIN**, która pozwala na łączenie danych z różnych tabel na podstawie wspólnych kolumn (kluczy obcych).

Typy połączeń

- INNER JOIN
- LEFT JOIN, RIGHT JOIN
- FULL JOIN
- CROSS JOIN (iloczyn kartezjański)
- SELF JOIN

INNER JOIN

BD: https://dimon.work/kurs/testdb.sql

Publisher

id	name	address
1	Everyman"s Library	NY
2	Oxford University Press	NY
3	Grand Central Publishing	Washington
4	Simon & Schuster	Chicago
5	West Coast Publishing	Chicago

Book

id	title	ISBN	fk_publisher_id
1	The Diary of a Young Girl	199535566	1
2	Pride and Prejudice	9780307594006	1
3	To Kill a Mockingbird	0446310786	2
4	The Book of Gutsy Women	1501178415	2
5	War and Peace	1788886526	2

Inner Join no publisher_id

id	name	address	title	ISBN
1	Everyman"s Library	NY	The Diary of a Young Girl	199535566
1	Everyman"s Library	NY	Pride and Prejudice	9780307594006
2	Oxford University Press	NY	To Kill a Mockingbird	0446310786
2	Oxford University Press	NY	The Book of Gutsy Women	1501178415
2	Oxford University Press	NY	War and Peace	1788886526

LEFT, RIGHT OUTER JOIN – połaczenia zewnętrzne

Publisher		
id	name	address
1	Everyman"s Library	NY
2	Oxford University Press	NY
3	Grand Central Publishing	Washington
4	Simon & Schuster	Chicago
5	West Coast Publishing	Chicago

Book			
id	title	ISBN	fk_publisher_id
1	The Diary of a Young Girl	199535566	1
2	Pride and Prejudice	9780307594006	1
3	To Kill a Mockingbird	0446310786	2
4	The Book of Gutsy Women	1501178415	2
5	War and Peace	1788886526	2

Left Outer Join no publisher_id

id	name	address	title	ISBN
1	Everyman"s Library	NY	The Diary of a Young Girl	199535566
1	Everyman"s Library	NY	Pride and Prejudice	9780307594006
2	Oxford University Press	NY	To Kill a Mockingbird	0446310786
2	Oxford University Press	NY	The Book of Gutsy Women	1501178415
2	Oxford University Press	NY	War and Peace	1788886526
3	Grand Central Publishing	Washington	NULL	NULL
4	Simon & Schuster	Chicago	NULL	NULL
5	West Coast Publishing	Chicago	NULL	NULL

RIGHT JOIN

RIGHT OUTER JOIN – zamiast tego polecenia można zawsze korzystać z poprzedniego, zmieniając kolejność tabel

Publisher					Book			
id	name	address			id	title	ISBN	fk_publisher_id
1	Everyman"s Library	NY			1	The Diary of a Young Girl	199535566	1
2	Oxford University Press	NY			2	Pride and Prejudice	9780307594006	1
3	Grand Central Publishing	Washington			3	To Kill a Mockingbird	0446310786	2
4	Simon & Schuster	Chicago			4	The Book of Gutsy Women	1501178415	2
5	West Coast Publishing	Chicago			5	War and Peace	1788886526	2

Right Outer Join no publisher_id

id	name	address	title	ISBN
1	Everyman"s Library	NY	The Diary of a Young Girl	199535566
1	Everyman"s Library	NY	Pride and Prejudice	9780307594006
2	Oxford University Press	NY	To Kill a Mockingbird	0446310786
2	Oxford University Press	NY	The Book of Gutsy Women	1501178415
2	Oxford University Press	NY	War and Peace	1788886526

FULL JOIN

FULL OUTER JOIN – LEFT JOIN + RIGHT JOIN

Publisher

id	name	address
1	Everyman"s Library	NY
2	Oxford University Press	NY
3	Grand Central Publishing	Washington
4	Simon & Schuster	Chicago
5	West Coast Publishing	Chicago

Book

id	title	ISBN	publisher_id
1	The Diary of a Young Girl	199535566	1
2	Pride and Prejudice	9780307594006	1
3	To Kill a Mockingbird	0446310786	2
4	The Book of Gutsy Women	1501178415	2
5	War and Peace	1788886526	2
6	The Man Who Didn't Call	5435955654	NULL

Full Outer Join no publisher_id

id	name	address	title	ISBN
1	Everyman"s Library	NY	The Diary of a Young Girl	199535566
1	Everyman"s Library	NY	Pride and Prejudice	9780307594006
2	Oxford University Press	NY	To Kill a Mockingbird	0446310786
2	Oxford University Press	NY	The Book of Gutsy Women	1501178415
2	Oxford University Press	NY	War and Peace	1788886526
3	Grand Central Publishing	Washington	NULL	NULL
4	Simon & Schuster	Chicago	NULL	NULL
NULL	NULL	NULL	The Man Who Didn't Call	5435955654

CROSS JOIN

CROSS JOIN –
(iloczyn kartezjański) każdy wiersz po lewej stronie jest dopasowywany do wszystkich rekordów po prawej stronie.

Cross Join

id	name	address	title
1	Everyman's Library	NY	The Diary of a Young Girl
1	Everyman's Library	NY	Pride and Prejudice
1	Everyman's Library	NY	To Kill a Mockingbird
1	Everyman's Library	NY	The Book of Gutsy Women: Favorite Stories of Courage and Resilience
1	Everyman's Library	NY	War and Peace
2	Oxford University Press	NY	The Diary of a Young Girl
2	Oxford University Press	NY	Pride and Prejudice
2	Oxford University Press	NY	To Kill a Mockingbird
2	Oxford University Press	NY	The Book of Gutsy Women: Favorite Stories of Courage and Resilience
2	Oxford University Press	NY	War and Peace
3	Grand Central Publishing	Washington	The Diary of a Young Girl
3	Grand Central Publishing	Washington	Pride and Prejudice
3	Grand Central Publishing	Washington	To Kill a Mockingbird
3	Grand Central Publishing	Washington	The Book of Gutsy Women: Favorite Stories of Courage and Resilience
3	Grand Central Publishing	Washington	War and Peace
4	Simon & Schuster	Chicago	The Diary of a Young Girl
4	Simon & Schuster	Chicago	Pride and Prejudice
4	Simon & Schuster	Chicago	To Kill a Mockingbird
4	Simon & Schuster	Chicago	The Book of Gutsy Women: Favorite Stories of Courage and Resilience
4	Simon & Schuster	Chicago	War and Peace
5	West Coast Publishing	Chicago	The Diary of a Young Girl
5	West Coast Publishing	Chicago	Pride and Prejudice
5	West Coast Publishing	Chicago	To Kill a Mockingbird
5	West Coast Publishing	Chicago	The Book of Gutsy Women: Favorite Stories of Courage and Resilience
5	West Coast Publishing	Chicago	War and Peace

EX. INNER JOIN

➔ Chcemy zobaczyć tytuły filmów, które są dostępne w magazynach (inventory).

```
SELECT film.title, inventory.inventory_id
-- Wybieramy tytuł filmu z tabeli film i identyfikator egzemplarza (inventory_id) z tabeli inventory.
FROM film
-- Zaczynamy od tabeli film, która zawiera informacje o filmach.
INNER JOIN inventory ON film.film_id = inventory.film_id;
-- Łączymy tabelę film z tabelą inventory, używając klucza film_id, który występuje w obu tabelach. Dzięki temu dostajemy wszystkie egzemplarze filmów dostępne w magazynach.
```

EX. INNER JOIN

➔ Chcemy zobaczyć tytuły filmów, które są dostępne w magazynach (inventory).

Ten wynik pokazuje, że filmy są dostępne w magazynie pod różnymi inventory_id w różnych sklepach.

title	inventory_id
ACADEMY DINOSAUR	1
ACADEMY DINOSAUR	2
ACADEMY DINOSAUR	3
ACADEMY DINOSAUR	4
ACADEMY DINOSAUR	5
ACADEMY DINOSAUR	6
ACADEMY DINOSAUR	7
ACADEMY DINOSAUR	8
ACE GOLDFINGER	9
ACE GOLDFINGER	10
ACE GOLDFINGER	11
ADAPTATION HOLES	12
Console N HOLES	13

EX. INNER JOIN

➔ Tym zapytaniem SQL chcemy uzyskać listę wszystkich filmów wypożyczonych przez konkretnego klienta (o customer_id = 1), wraz z datami wypożyczenia.

```
SELECT customer.first_name, customer.last_name, film.title, rental.rental_date
FROM rental

INNER JOIN customer ON rental.customer_id = customer.customer_id
INNER JOIN inventory ON rental.inventory_id = inventory.inventory_id
INNER JOIN film ON inventory.film_id = film.film_id

WHERE customer.customer_id = 1;
```

Powyższe zapytanie wybiera imię i nazwisko klienta, tytuł wypożyczonego filmu oraz datę wypożyczenia dla klienta o ID 1.

Wynikiem tego zapytania będzie lista wypożyczeń dla konkretnego klienta (o customer_id = 1), pokazująca imię i nazwisko tego klienta, tytuły wypożyczonych filmów oraz daty wypożyczeń.

EX. INNER JOIN

1. Zapytanie zaczyna od tabeli rental i szuka wypożyczeń.
2. Następnie łączy te wypożyczenia z tabelą customer, aby dowiedzieć się, kto wypożyczył film.
3. Dalej łączy dane wypożyczeń z tabelą inventory, aby uzyskać informacje o egzemplarzach filmów, które były wypożyczone.
4. Na końcu łączy te dane z tabelą film, aby wyświetlić tytuł filmu.
5. Całość jest ograniczona do wypożyczeń dokonanych przez klienta o customer_id = 1.

EX. **INNER JOIN**

first_name	last_name	title	rental_date
MARY	SMITH	PATIENT SISTER	2005-05-25 11:30:37
MARY	SMITH	TALENTED HOMICIDE	2005-05-28 10:35:23
MARY	SMITH	MUSKETEERS WAIT	2005-06-15 00:54:12
MARY	SMITH	DETECTIVE VISION	2005-06-15 18:02:53
MARY	SMITH	FERRIS MOTHER	2005-06-15 21:08:46
MARY	SMITH	CLOSER BANG	2005-06-16 15:18:57
MARY	SMITH	ATTACKS HATE	2005-06-18 08:41:48
MARY	SMITH	SAVANNAH TOWN	2005-06-18 13:33:59

EX. INNER JOIN

Zadania:

- **Zadanie 1:** Napisz zapytanie, które wyświetli listę wszystkich filmów wypożyczonych przez klientów o imieniach "BARBARA ", "MARY " wraz z datami wypożyczeń.
- **Zadanie 2:** Napisz zapytanie, które pokaże listę filmów oraz ich kategorii (category) wypożyczonych w sklepie o ID 2.
- **Zadanie 3:** Znajdź pracownika, który obsłużył najwięcej wypożyczeń, wyświetlając jego imię, nazwisko oraz liczbę wypożyczeń.

EX. LEFT JOIN

Przykład **Klienci** i **płatności**:

Chcemy uzyskać listę wszystkich klientów oraz ich płatności. Jeśli klient nie dokonał żadnych płatności, nadal będzie widoczny w wynikach, a pole płatności będzie miało wartość NULL.

```
SELECT customer.first_name, customer.last_name, payment.amount
-- Wybieramy imię i nazwisko klientów oraz kwotę płatności.
FROM customer
-- Rozpoczynamy od tabeli customer, co oznacza, że to ona jest tabelą po lewej stronie.
LEFT JOIN payment ON customer.customer_id = payment.customer_id;
-- Łączymy tabelę customer z tabelą payment na podstawie customer_id. Dzięki temu uzyskujemy wszystkie wiersze z tabeli customer, nawet jeśli nie mają one przypisanych płatności w tabeli payment.
```

EX. LEFT JOIN

Ile w sumie każdy zapłacił ?

```
SELECT customer.first_name, customer.last_name,
SUM(payment.amount) AS total_amount
-- Używamy SUM(payment.amount) do zsumowania płatności dla każdego klienta.
FROM customer
LEFT JOIN payment ON customer.customer_id = payment.customer_id
GROUP BY customer.customer_id
-- Grupujemy wyniki według customer_id, co pozwala na obliczenie sumy płatności dla każdego klienta.
ORDER BY total_amount DESC;
-- Sortujemy wyniki według łącznej kwoty płatności w kolejności malejącej.
```

EX. LEFT JOIN

Ile w sumie każdy zapłacił ?

first_name	last_name	total_amount ▼ 1
KARL	SEAL	221.55
ELEANOR	HUNT	216.54
CLARA	SHAW	195.58
RHONDA	KENNEDY	194.61
MARION	SNYDER	194.61
TOMMY	COLLAZO	186.62
WESLEY	BULL	177.60
TIM	CARY	175.61
MARCIA	DEAN	175.58
ANA	BRADLEY	174.66

EX. LEFT JOIN

Wyświetlenie wszystkich filmów i ich kategorii

```
SELECT film.title AS Film, category.name AS Kategoria
FROM film
LEFT JOIN film_category ON film.film_id = film_category.film_id
LEFT JOIN category
ON film_category.category_id = category.category_id;
```

Film	Kategoria
ACADEMY DINOSAUR	Documentary
ACE GOLDFINGER	Horror
ADAPTATION HOLES	Documentary

EX. LEFT JOIN

Zadania:

- **Zadanie 1:** Wyświetlenie wszystkich płatności i związanych z nimi klientów

  Opis: Chcemy zobaczyć wszystkie płatności oraz klientów, którzy je dokonali. W przypadku płatności, które nie mają przypisanego klienta, chcemy je również wyświetlić.
- **Zadanie 2:** Wyświetlenie wszystkich pracowników i ich filmów

  Opis: Chcemy zobaczyć wszystkich pracowników oraz filmy, które wypożyczali. W przypadku pracowników, którzy nie obsługiwali żadnych filmów, ich dane również powinny być wyświetlone.
- **Zadanie 3:** Wyświetlenie wszystkich klientów i ich wypożyczeń

  Opis: Chcemy zobaczyć wszystkich klientów oraz wypożyczenia, które zrealizowali. Jeśli klient nie ma żadnych wypożyczeń, jego dane również powinny być wyświetlone.

EX. LEFT JOIN

Zadania:

- Wstaw rekord reprezentujący Mary Smith wypożyczającą dziś film „Academy Dinosaur” od Mike'a Hillyera w Store 1.
- Kiedy będzie można obejrzeć film „Academy Dinosaur”?
- Jaki jest średni czas trwania wszystkich filmów w sakila DB?
- Jaki jest średni czas trwania filmów według kategorii?
- Dlaczego to zapytanie zwraca pusty zbiór?

EX. FULL JOIN (LEFT+RIGHT)

Zadanie: Połączyć tabele actor i film_actor, aby wyświetlić wszystkich aktorów i ich filmy, w tym aktorów, którzy nie wystąpili w żadnym filmie i filmy, które mogą nie mieć informacji o aktorze.

1. LEFT JOIN: Wybieramy wszystkich aktorów z tabeli actor, a także informacje o filmie, jeśli aktor wystąpił w jakimkolwiek filmie. Jeśli aktor nie wystąpił w żadnym filmie, informacje o filmie będą miały wartość NULL.
2. RIGHT JOIN: Wybieramy wszystkie filmy z tabeli film_actor i odpowiadających im aktorów. Jeśli nie ma informacji o aktorze, dane aktora będą miały wartość NULL.
3. UNION: Ten operator łączy wyniki obu zapytań, tworząc odpowiednik FULL JOIN, w którym wyprowadzane są wszystkie rekordy z obu tabel.

EX. FULL JOIN (LEFT+RIGHT)


```
SELECT actor.actor_id, actor.first_name, actor.last_name,
film_actor.film_id
FROM actor
LEFT JOIN film_actor ON actor.actor_id = film_actor.actor_id

UNION

SELECT actor.actor_id, actor.first_name, actor.last_name,
film_actor.film_id
FROM actor
RIGHT JOIN film_actor ON actor.actor_id = film_actor.actor_id;
```

1	PENELOPE	GUINESS	832
1	PENELOPE	GUINESS	939
1	PENELOPE	GUINESS	970
1	PENELOPE	GUINESS	980
2	NICK	WAHLBERG	3
2	NICK	WAHLBERG	31
2	NICK	WAHLBERG	47
2	NICK	WAHLBERG	105
2	NICK	WAHLBERG	132
2	NICK	WAHLBERG	145

SELF JOIN

SELF JOIN to złączenie tabeli z samą sobą. Jest to przydatne, gdy chcesz porównać lub dopasować wiersze w tej samej tabeli. Jest on stosowany w taki sam sposób jak zwykły JOIN, ale tabela jest przywoływana dwukrotnie pod różnymi aliasami.

⬇ Chcemy porównać podmioty według daty ostatniej aktualizacji ich danych.

EX. SELF JOIN


```
SELECT
    actor1.actor_id AS Actor_ID_1,
    actor1.first_name AS First_Name_1,
    actor2.actor_id AS Actor_ID_2,
    actor2.first_name AS First_Name_2,
    actor1.last_update
FROM actor AS actor1
-- Tutaj „actor1” jest aliasem dla tabeli „actor”
JOIN actor AS actor2
-- Tutaj „actor2” jest drugim aliasem dla tej samej tabeli „actor”
    ON actor1.last_update = actor2.last_update
    AND actor1.actor_id <> actor2.actor_id;
```

EX. SELF JOIN

Wyjaśnienie:

1. SELF JOIN łączy tabelę aktorów z nią samą.
   actor1.last_update = actor2.last_update - dopasowujemy aktorów, których dane zostały zaktualizowane w tym samym czasie.
2. actor1.actor_id <> actor2.actor_id - wykluczamy porównanie jednego aktora z samym sobą.
3. Pary aktorów z tą samą datą aktualizacji, ale różnymi identyfikatorami są wyprowadzane.

To zapytanie demonstruje proste auto-łączenie w celu znalezienia identycznych wartości w tej samej tabeli.