

## Bazy danych 2019

na podstawie slajdów Przemysławy Kanarek

21 lutego 2019

# BD i SZBD

# BD i SZBD

Baza danych (BD,DB) — zbiór danych zawierających zarówno informacje rzeczowe, jak i strukturę tych informacji; zazwyczaj **duży**, **długotrwały**, dostępny **dla wielu użytkowników** na różne sposoby;

# BD i SZBD

**Baza danych (BD,DB)** — zbiór danych zawierających zarówno informacje rzeczowe, jak i strukturę tych informacji; zazwyczaj **duży**, **długotrwały**, dostępny **dla wielu użytkowników** na różne sposoby;

**System Zarządzania Bazami Danych (SZBD, DBMS)** — oprogramowanie pozwalające **definiować** strukturę bazy danych, **gromadzić dane** w bazie i je **efektywnie udostępniać** (Oracle, PostgreSQL, MySQL, SQL Server, DB2,...);

# BD i SZBD

**Baza danych (BD,DB)** — zbiór danych zawierających zarówno informacje rzeczowe, jak i strukturę tych informacji; zazwyczaj **duży**, **długotrwały**, dostępny **dla wielu użytkowników** na różne sposoby;

**System Zarządzania Bazami Danych (SZBD, DBMS)** — oprogramowanie pozwalające **definiować** strukturę bazy danych, **gromadzić dane** w bazie i je **efektywnie udostępniać** (Oracle, PostgreSQL, MySQL, SQL Server, DB2,...);

**System bazy danych** — baza danych założona i użytkowana pod konkretnym SZBD;

# BD i SZBD

**Baza danych (BD,DB)** — zbiór danych zawierających zarówno informacje rzeczowe, jak i strukturę tych informacji; zazwyczaj **duży**, **długotrwały**, dostępny **dla wielu użytkowników** na różne sposoby;

**System Zarządzania Bazami Danych (SZBD, DBMS)** — oprogramowanie pozwalające **definiować** strukturę bazy danych, **gromadzić dane** w bazie i je **efektywnie udostępniać** (Oracle, PostgreSQL, MySQL, SQL Server, DB2,...);

**System bazy danych** — baza danych założona i użytkowana pod konkretnym SZBD;

**ACID** — atomowość (Atomic), poprawność (Consistent), niezależność (Independent), trwałość (Durable).

# BD i SZBD

**Baza danych (BD,DB)** — zbiór danych zawierających zarówno informacje rzeczowe, jak i strukturę tych informacji; zazwyczaj **duży**, **długotrwały**, dostępny **dla wielu użytkowników** na różne sposoby;

**System Zarządzania Bazami Danych (SZBD, DBMS)** — oprogramowanie pozwalające **definiować** strukturę bazy danych, **gromadzić dane** w bazie i je **efektywnie udostępniać** (Oracle, PostgreSQL, MySQL, SQL Server, DB2,...);

**System bazy danych** — baza danych założona i użytkowana pod konkretnym SZBD;

**ACID** — atomowość (Atomic), poprawność (Consistent), niezależność (Independent), trwałość (Durable).

**Security** —kontrola dostępu

# Języki baz danych



# Języki baz danych

Diagramy E-R, UML — projektowanie konceptualne (modelowanie);

# Języki baz danych

Diagramy E-R, UML — projektowanie konceptualne (modelowanie);

Język definiowania danych (DDL) — polecenia tworzenia elementów struktury bazy danych;

# Języki baz danych

Diagramy E-R, UML — projektowanie konceptualne (modelowanie);

Język definiowania danych (DDL) — polecenia tworzenia elementów struktury bazy danych;

Język zapytań (query language) — polecenia wyszukiwania danych;

# Języki baz danych

Diagramy E-R, UML — projektowanie konceptualne (modelowanie);

Język definiowania danych (DDL) — polecenia tworzenia elementów struktury bazy danych;

Język zapytań (query language) — polecenia wyszukiwania danych;

Język modyfikacji danych (DML) — polecenia dodawania, usuwania i modyfikacji danych;

# Języki baz danych

Diagramy E-R, UML — projektowanie konceptualne (modelowanie);

Język definiowania danych (DDL) — polecenia tworzenia elementów struktury bazy danych;

Język zapytań (query language) — polecenia wyszukiwania danych;

Język modyfikacji danych (DML) — polecenia dodawania, usuwania i modyfikacji danych;

Język aplikacji — język służący do pisania aplikacji odwołujących się do bazy danych.

# Języki baz danych

Diagramy E-R, UML — projektowanie konceptualne (modelowanie);

Język definiowania danych (DDL) — polecenia tworzenia elementów struktury bazy danych;

Język zapytań (query language) — polecenia wyszukiwania danych;

Język modyfikacji danych (DML) — polecenia dodawania, usuwania i modyfikacji danych;

Język aplikacji — język służący do pisania aplikacji odwołujących się do bazy danych.

**SQL (Structured Query Language)** zawiera DDL, query language oraz DML i jest zaimplementowany praktycznie we wszystkich relacyjnych SZBD (*dialekty SQL*).

**Języki aplikacji** mogą to być języki programowania, dla których zaprogramowano biblioteki dostępu do bazy danych ("naśladujące" polecenia SQL), własny język programowania SZBD stanowiący rozszerzenie SQL lub programistycznego SQL.

# Plan wykładu

# Plan wykładu

## Będzie o:

- 1 **Model relacyjny teoretycznie:** elementy składowe modelu, języki zapytań, postaci normalne (BCNF, 3NF, 4NF).
- 2 **Model relacyjny praktycznie:** zapytania SQL, projektowanie baz danych oraz diagramy E-R i UML, język definicji danych SQL
- 3 **Systemy zarządzania relacyjnymi bazami danych:** przetwarzanie zapytań, transakcje i wielodostęp, bezpieczeństwo danych, struktury dostępu do danych.



# Plan wykładu

## Będzie o:

- 1 **Model relacyjny teoretycznie:** elementy składowe modelu, języki zapytań, postaci normalne (BCNF, 3NF, 4NF).
- 2 **Model relacyjny praktycznie:** zapytania SQL, projektowanie baz danych oraz diagramy E-R i UML, język definicji danych SQL
- 3 **Systemy zarządzania relacyjnymi bazami danych:** przetwarzanie zapytań, transakcje i wielodostęp, bezpieczeństwo danych, struktury dostępu do danych.

# Plan wykładu

## Będzie o:

- 1 **Model relacyjny teoretycznie:** elementy składowe modelu, języki zapytań, postaci normalne (BCNF, 3NF, 4NF).
- 2 **Model relacyjny praktycznie:** zapytania SQL, projektowanie baz danych oraz diagramy E-R i UML, język definicji danych SQL
- 3 **Systemy zarządzania relacyjnymi bazami danych:** przetwarzanie zapytań, transakcje i wielodostęp, bezpieczeństwo danych, struktury dostępu do danych.

## Na innych przedmiotach:

- Bazy grafowe
- MapReduce, Bazy rozproszone, noSQL, newSQL
- DataMining i hurtownie danych
- DBMS
- Bazy geograficzne, mobilne
- Bazy strumieniowe, analityczne, kolumnowe

# Dlaczego relacyjne bazy danych?

# Dlaczego relacyjne bazy danych?

## Jak do tego doszło?

- **Lata 50-60-te:** powstaje model hierarchiczny (IMS) i sieciowy (CODASYL).

# Dlaczego relacyjne bazy danych?

## Jak do tego doszło?

- **Lata 50-60-te:** powstaje model hierarchiczny (IMS) i sieciowy (CODASYL).
- **Lata 70-te:** Codd proponuje model relacyjny i powstają pierwsze relacyjne SZBD: Ingres (Ingres Corp, PostgreSQL, Sybase, MS SQL Server, ...) oraz System R (DB2, Oracle,...).

# Dlaczego relacyjne bazy danych?

## Jak do tego doszło?

- **Lata 50-60-te:** powstaje model hierarchiczny (IMS) i sieciowy (CODASYL).
- **Lata 70-te:** Codd proponuje model relacyjny i powstają pierwsze relacyjne SZBD: Ingres (Ingres Corp, PostgreSQL, Sybase, MS SQL Server, ...) oraz System R (DB2, Oracle,...).
- **Lata 90-te:** model relacyjny rządzi, ale...

# Dlaczego relacyjne bazy danych?

## Jak do tego doszło?

- **Lata 50-60-te:** powstaje model hierarchiczny (IMS) i sieciowy (CODASYL).
- **Lata 70-te:** Codd proponuje model relacyjny i powstają pierwsze relacyjne SZBD: Ingres (Ingres Corp, PostgreSQL, Sybase, MS SQL Server, ...) oraz System R (DB2, Oracle,...).
- **Lata 90-te:** model relacyjny rządzi, ale... staje się za ciasny, bo nie zawsze dobrze sobie radzi ze skomplikowanymi danymi, specyficznym przetwarzaniem danych i funkcjonowaniem w nowych środowiskach (chmura, urządzenia mobline, dyski SSD).

# Dlaczego relacyjne bazy danych?

## Jak do tego doszło?

- **Lata 50-60-te:** powstaje model hierarchiczny (IMS) i sieciowy (CODASYL).
- **Lata 70-te:** Codd proponuje model relacyjny i powstają pierwsze relacyjne SZBD: Ingres (Ingres Corp, PostgreSQL, Sybase, MS SQL Server, ...) oraz System R (DB2, Oracle,...).
- **Lata 90-te:** model relacyjny rządzi, ale... staje się za ciasny, bo nie zawsze dobrze sobie radzi ze skomplikowanymi danymi, specyficznym przetwarzaniem danych i funkcjonowaniem w nowych środowiskach (chmura, urządzenia mobilne, dyski SSD).

*Jack of all trades, master of none*



# Dlaczego relacyjne bazy danych?

## Jak do tego doszło?

- **Lata 50-60-te:** powstaje model hierarchiczny (IMS) i sieciowy (CODASYL).
- **Lata 70-te:** Codd proponuje model relacyjny i powstają pierwsze relacyjne SZBD: Ingres (Ingres Corp, PostgreSQL, Sybase, MS SQL Server, ...) oraz System R (DB2, Oracle,...).
- **Lata 90-te:** model relacyjny rządzi, ale... staje się za ciasny, bo nie zawsze dobrze sobie radzi ze skomplikowanymi danymi, specyficznym przetwarzaniem danych i funkcjonowaniem w nowych środowiskach (chmura, urządzenia mobilne, dyski SSD).









*Jack of all trades, master of none*

## XXI wiek

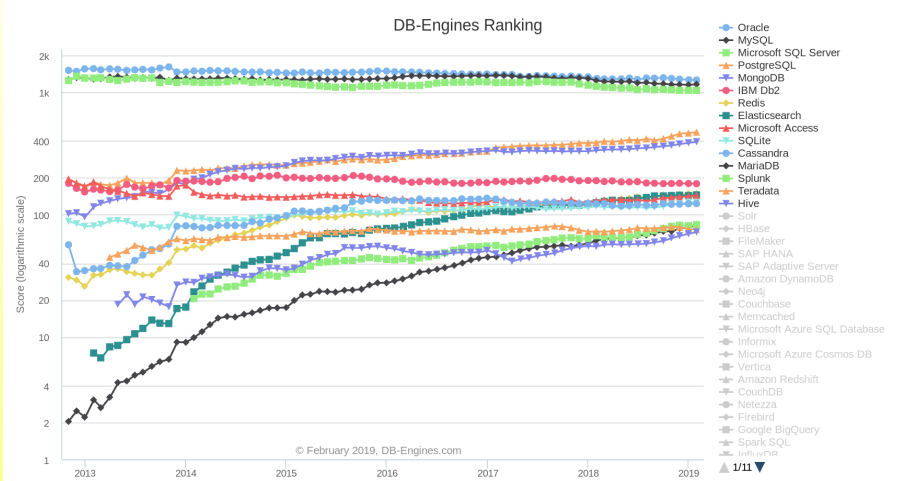
- NoSQL → NewSQL...
- Powstają nowe modele danych: obiektowy, semistrukturalny (XML), astrukturalny (BigTable/HBase),...
- Powstają specjalistyczne systemy baz danych: temporalne, probabilistyczne, geograficzne, tekstowe, grafowe,...
- Powstają bazy dostosowane do nowych nośników: mobilne, SSD,...
- Bazy są większe niż kiedykolwiek wcześniej: skalowalność, przetwarzanie analityczne (hurtownie danych), przetwarzanie strumieniowe,...
- Zasady ACID (czasem) nie są kluczowe: bazy sieci społecznościowych (mniejsza niezawodność), informacji (opóźniona spójność),...

# Popularność: <https://db-engines.com/en/ranking>

343 systems in ranking, February 2019

Rank			DBMS	Database Model	Score		
Feb 2019	Jan 2019	Feb 2018			Feb 2019	Jan 2019	Feb 2018
1.	1.	1.	Oracle +	Relational, Multi-model 	1264.02	-4.82	-39.26
2.	2.	2.	MySQL +	Relational, Multi-model 	1167.29	+13.02	-85.18
3.	3.	3.	Microsoft SQL Server +	Relational, Multi-model 	1040.05	-0.21	-81.98
4.	4.	4.	PostgreSQL +	Relational, Multi-model 	473.56	+7.45	+85.18
5.	5.	5.	MongoDB +	Document	395.09	+7.91	+58.67
6.	6.	6.	IBM Db2 +	Relational, Multi-model 	179.42	-0.43	-10.55
7.	7.	↑ 8.	Redis +	Key-value, Multi-model 	149.45	+0.43	+22.43
8.	8.	↑ 9.	Elasticsearch +	Search engine, Multi-model 	145.25	+1.81	+19.93
9.	9.	↓ 7.	Microsoft Access	Relational	144.02	+2.41	+13.95
10.	10.	↑ 11.	SQLite +	Relational	126.17	-0.63	+8.89
11.	11.	↓ 10.	Cassandra +	Wide column	123.37	+0.39	+0.59
12.	↑ 13.	↑ 17.	MariaDB +	Relational, Multi-model 	83.42	+4.60	+21.77
13.	↓ 12.	13.	Splunk	Search engine	82.81	+1.39	+15.55
14.	14.	↓ 12.	Teradata +	Relational	75.97	-0.22	+2.98

Trendy: [https://db-engines.com/en/ranking\\_trend](https://db-engines.com/en/ranking_trend)



## Ostrzeżenie:

<http://avid.cs.umass.edu/courses/691LL/f2006/papers/SH05.pdf>

# What Goes Around Comes Around

Michael Stonebraker  
Joseph M. Hellerstein

## Abstract

This paper provides a summary of 35 years of data model proposals, grouped into 9 different eras. We discuss the proposals of each era, and show that there are only a few basic data modeling ideas, and most have been around a long time. Later proposals inevitably bear a strong resemblance to certain earlier proposals. Hence, it is a worthwhile exercise to study previous proposals.

## Ostrzeżenie:

<http://avid.cs.umass.edu/courses/691LL/f2006/papers/SH05.pdf>

# What Goes Around Comes Around

Most everything put forward in the last 20 years is  
a reinvention of something from a quarter century ago!

## Abstract

This paper provides a summary of 35 years of data model proposals, grouped into 9 different eras. We discuss the proposals of each era, and show that there are only a few basic data modeling ideas, and most have been around a long time. Later proposals inevitably bear a strong resemblance to certain earlier proposals. Hence, it is a worthwhile exercise to study previous proposals.

# Ćwiczenia i pracownia

## Będziemy się uczyć:

- 1 Rozumieć model relacyjny (algebra relacji i rachunki relacyjne, postaci normalne);
- 2 Korzystać z gotowej bazy danych — wyszukiwać w niej informacje, odpowiedzi na interesujące nas pytania (język SQL);
- 3 Konstruować poprawne bazy danych dla zagadnień rzeczywistych — projektować bazy (modelować) i na podstawie projektów definiować elementy baz danych;
- 4 Tworzyć aplikacje korzystające z bazy danych.

# Ćwiczenia i pracownia

## Będziemy się uczyć:

- 1 Rozumieć model relacyjny (algebra relacji i rachunki relacyjne, postaci normalne);
- 2 Korzystać z gotowej bazy danych — wyszukiwać w niej informacje, odpowiedzi na interesujące nas pytania (język SQL);
- 3 Konstruować poprawne bazy danych dla zagadnień rzeczywistych — projektować bazy (modelować) i na podstawie projektów definiować elementy baz danych;
- 4 Tworzyć aplikacje korzystające z bazy danych.

Materiały i informacje: [skos.ii.uni.wroc.pl](http://skos.ii.uni.wroc.pl) — kurs Bazy Danych 2019.

# Literatura

- Jeffrey D. Ullman, Jennifer Widom, Podstawowy Kurs Systemów Baz Danych, WNT, Warszawa 1999;
- Garcia-Molina H., Ullman J.D., Widom J., Implementacja systemów baz danych, WNT, 2003 (seria: Klasyka Informatyki);
- Garcia-Molina H., Ullman J.D., Widom J., Database Systems: The Complete Book (suma dwóch powyższych pozycji);
- Thomas Connolly, Carolyn Begg, Database Systems, Addison Wesley 2002, także po polsku: ReadMe 2004;
- Date C. J., An Introduction to Database System, vol. II, Addison-Wesley Pub. Comp., również WNT W-wa, (seria: Klasyka Informatyki), 2000;
- R. Ramakrishnan, J. Gehrke, Database Management Systems, 2nd edition, WCB/McGraw-Hill, 2001. Jest też wydanie 3-cie.



# Elementy modelu

# Elementy modelu

**Relacja (tabela)** — jedyna struktura dla danych w modelu; ma ustaloną liczbę kolumn, w które można wpisywać wartości ustalonego typu i dowolną liczbę wierszy.

# Elementy modelu

**Relacja (tabela)** — jedyna struktura dla danych w modelu; ma ustaloną liczbę kolumn, w które można wpisywać wartości ustalonego typu i dowolną liczbę wierszy.

**Więzy (warunki poprawności, warunki spójności)** — dane wpisywane do tabel muszą spełniać zdefiniowane warunki: typ danych, zakres,...

# Elementy modelu

- Relacja (tabela)** — jedyna struktura dla danych w modelu; ma ustaloną liczbę kolumn, w które można wpisywać wartości ustalonego typu i dowolną liczbę wierszy.
- Więzy (warunki poprawności, warunki spójności)** — dane wpisywane do tabel muszą spełniać zdefiniowane warunki: typ danych, zakres,...
- Baza danych** — zbiór tabel z danymi spełniającymi nałożone na nie więzy.

# Elementy modelu

**Relacja (tabela)** — jedyna struktura dla danych w modelu; ma ustaloną liczbę kolumn, w które można wpisywać wartości ustalonego typu i dowolną liczbę wierszy.

**Więzy (warunki poprawności, warunki spójności)** — dane wpisywane do tabel muszą spełniać zdefiniowane warunki: typ danych, zakres,...

**Baza danych** — zbiór tabel z danymi spełniającymi nałożone na nie więzy.

**Język zapytań (*query language*)** — algebra relacji, relacyjny rachunek krotek i relacyjny rachunek dziedzin — formalne języki pozwalające wyszukać w relacjach określoną informację.

# Relacja, czyli tabela

## Osoba

Nazwisko : varchar(20)	PESEL : char(11)	dataUr : date
...		
Abacki	80121304455	'20-02-1980'
...		

## Mieszkanie

PESEL : char(11)	Adres : varchar(50)	Metraż : real
...		
80121304455	Elk, Kwiatowa 100	60,2
80121304455	Poznań, Szeroka 10/2	30,2
NULL	Elk, Kwiatowa 102	64,2
...		

## Elementy relacji

- *Atrybut* — nazwa kolumny;
- *Dziedzina* — typ danych;
- *Krotność (arność)* — liczba atrybutów;
- *Krotka (wiersz)* — element relacji;
- *Atrybuty krotki* — *Osoba*[3] lub *Mieszkanie*.Adres;
- *Schemat relacji* — nazwa relacji, nazwy i typy kolumn;
- *Stan relacji* to zawarte w niej krotki.

# Relacja, czyli tabela

## Osoba

Nazwisko : varchar(20)	PESEL : char(11)	dataUr : date
...		
Abacki	80121304455	'20-02-1980'
...		

## Mieszkanie

PESEL : char(11)	Adres : varchar(50)	Metraż : real
...		
80121304455	Elk, Kwiatowa 100	60,2
80121304455	Poznań, Szeroka 10/2	30,2
NULL	Elk, Kwiatowa 102	64,2
...		

## Elementy relacji

- *Atrybut* — nazwa kolumny;
- *Dziedzina* — typ danych;
- *Krotność (arność)* — liczba atrybutów;
- *Krotka (wiersz)* — element relacji;
- *Atrybuty krotki* — *Osoba*[3] lub *Mieszkanie*.Adres;
- *Schemat relacji* — nazwa relacji, nazwy i typy kolumn;
- *Stan relacji* to zawarte w niej krotki.

# Relacja, czyli tabela

## Osoba

Nazwisko : <b>varchar(20)</b>	PESEL : <b>char(11)</b>	dataUr : <b>date</b>
...		
Abacki	80121304455	'20-02-1980'
...		

## Mieszkanie

PESEL : <b>char(11)</b>	Adres : <b>varchar(50)</b>	Metraż : <b>real</b>
...		
80121304455	Elk, Kwiatowa 100	60,2
80121304455	Poznań, Szeroka 10/2	30,2
NULL	Elk, Kwiatowa 102	64,2
...		

## Elementy relacji

- *Atrybut* — nazwa kolumny;
- *Dziedzina* — typ danych;
- *Krotność (arność)* — liczba atrybutów;
- *Krotka (wiersz)* — element relacji;
- *Atrybuty krotki* — *Osoba*[3] lub *Mieszkanie*.Adres;
- *Schemat relacji* — nazwa relacji, nazwy i typy kolumn;
- *Stan relacji* to zawarte w niej krotki.



# Relacja, czyli tabela

## Osoba

Nazwisko : varchar(20)	PESEL : char(11)	dataUr : date
...		
Abacki	80121304455	'20-02-1980'
...		

## Mieszkanie

PESEL : char(11)	Adres : varchar(50)	Metraż : real
...		
80121304455	Elk, Kwiatowa 100	60,2
80121304455	Poznań, Szeroka 10/2	30,2
NULL	Elk, Kwiatowa 102	64,2
...		

## Elementy relacji

- *Atrybut* — nazwa kolumny;
- *Dziedzina* — typ danych;
- *Krotność (arność)* — **liczba atrybutów**;
- *Krotka (wiersz)* — element relacji;
- *Atrybuty krotki* — *Osoba*[3] lub *Mieszkanie*.Adres;
- *Schemat relacji* — nazwa relacji, nazwy i typy kolumn;
- *Stan relacji* to zawarte w niej krotki.

# Relacja, czyli tabela

## Osoba

Nazwisko : varchar(20)	PESEL : char(11)	dataUr : date
...	80121304455	'20-02-1980'
Abacki		
...		

## Mieszkanie

PESEL : char(11)	Adres : varchar(50)	Metraż : real
...		
80121304455	Elk, Kwiatowa 100	60,2
80121304455	Poznań, Szeroka 10/2	30,2
NULL	Elk, Kwiatowa 102	64,2
...		

## Elementy relacji

- *Atrybut* — nazwa kolumny;
- *Dziedzina* — typ danych;
- *Krotność (arność)* — liczba atrybutów;
- *Krotka (wiersz)* — element relacji;
- *Atrybuty krotki* — *Osoba*[3] lub *Mieszkanie*.Adres;
- *Schemat relacji* — nazwa relacji, nazwy i typy kolumn;
- *Stan relacji* to zawarte w niej krotki.

# Relacja, czyli tabela

## Osoba

Nazwisko : varchar(20)	PESEL : char(11)	dataUr : date
...		
Abacki	80121304455	'20-02-1980'
...		

## Mieszkanie

PESEL : char(11)	Adres : varchar(50)	Metraż : real
...		
80121304455	Elk, Kwiatowa 100	60,2
80121304455	Poznań, Szeroka 10/2	30,2
NULL	Elk, Kwiatowa 102	64,2
...		

## Elementy relacji

- *Atrybut* — nazwa kolumny;
- *Dziedzina* — typ danych;
- *Krotność (arność)* — liczba atrybutów;
- *Krotka (wiersz)* — element relacji;
- *Atrybuty krotki* — *Osoba*[3] lub *Mieszkanie*.Adres;
- *Schemat relacji* — nazwa relacji, nazwy i typy kolumn;
- *Stan relacji* to zawarte w niej krotki.

# Relacja, czyli tabela

## Osoba

Nazwisko : varchar(20)	PESEL : char(11)	dataUr : date
...		
Abacki	80121304455	'20-02-1980'
...		

## Mieszkanie

PESEL : char(11)	Adres : varchar(50)	Metraż : real
...		
80121304455	Elk, Kwiatowa 100	60,2
80121304455	Poznań, Szeroka 10/2	30,2
NULL	Elk, Kwiatowa 102	64,2
...		

## Elementy relacji

- *Atrybut* — nazwa kolumny;
- *Dziedzina* — typ danych;
- *Krotność (arność)* — liczba atrybutów;
- *Krotka (wiersz)* — element relacji;
- *Atrybuty krotki* — *Osoba*[3] lub *Mieszkanie*.Adres;
- *Schemat relacji* — nazwa relacji, nazwy i typy kolumn;
- *Stan relacji* to zawarte w niej krotki.

# Relacja, czyli tabela

## Osoba

Nazwisko : varchar(20)	PESEL : char(11)	dataUr : date
...	80121304455	'20-02-1980'
Abacki		
...		

## Mieszkanie

PESEL : char(11)	Adres : varchar(50)	Metraż : real
...		
80121304455	Elk, Kwiatowa 100	60,2
80121304455	Poznań, Szeroka 10/2	30,2
NULL	Elk, Kwiatowa 102	64,2
...		

## Elementy relacji

- *Atrybut* — nazwa kolumny;
- *Dziedzina* — typ danych;
- *Krotność (arność)* — liczba atrybutów;
- *Krotka (wiersz)* — element relacji;
- *Atrybuty krotki* — *Osoba*[3] lub *Mieszkanie*.Adres;
- *Schemat relacji* — nazwa relacji, nazwy i typy kolumn;
- *Stan relacji* to zawarte w niej krotki.

# Notacja matematyczna

Dla atrybutów  $A_1, \dots, A_k$  i związanych z nimi dziedzin  $D_1, \dots, D_k$  relacja  $R$  ma:

**schemat**  $R = A_1 \dots A_k$  lub  $R(A_1, \dots, A_k)$ ,

**arność**  $k$ ,

**stan**  $r \subseteq D_1 \times \dots \times D_k$ ,

**krotki**  $(v_1, v_2, \dots, v_k) \in r$ .

**Relacyjna baza danych (schemat i stan)** to zbiór relacji o różnych nazwach.

# Notacja matematyczna

Dla atrybutów  $A_1, \dots, A_k$  i związanych z nimi dziedzin  $D_1, \dots, D_k$  relacja  $R$  ma:

**schemat**  $R = A_1 \dots A_k$  lub  $R(A_1, \dots, A_k)$ ,

**arność**  $k$ ,

**stan**  $r \subseteq D_1 \times \dots \times D_k$ ,

**krotki**  $(v_1, v_2, \dots, v_k) \in r$ .

**Relacyjna baza danych (schemat i stan)** to zbiór relacji o różnych nazwach.

W przykładzie:

- Osoba(Nazwisko,PESEL,dataUr),
- Mieszkanie(PESEL,Adres,Metraż)

# Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL	Elk, Kwiatowa 102	64,2	$\Leftarrow t_2$
NULL			
...			



# Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL	Elk, Kwiatowa 102	64,2	$\Leftarrow t_2$
NULL			
...			

## Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...	Poznań, Szeroka 10/12 Elk, Kwiatowa 102	64,2	$\Leftarrow t_1$
NULL NULL ...		64,2	$\Leftarrow t_2$

- $t_1.\text{PESEL} = t_2.\text{PESEL}$

# Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...	Poznań, Szeroka 10/12 Elk, Kwiatowa 102	64,2	$\Leftarrow t_1$
NULL NULL ...		64,2	$\Leftarrow t_2$

- $t_1.\text{PESEL} = t_2.\text{PESEL}$  UNKNOWN!!!

# Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL	Elk, Kwiatowa 102	64,2	$\Leftarrow t_2$
...			

- $t_1$ .PESEL =  $t_2$ .PESEL UNKNOWN!!!
- $t_1$ .PESEL =  $t_1$ .Adres

# Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL	Elk, Kwiatowa 102	64,2	$\Leftarrow t_2$
...			

- $t_1$ .PESEL =  $t_2$ .PESEL UNKNOWN!!!
- $t_1$ .PESEL =  $t_1$ .Adres UNKNOWN!!!

# Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...			
NULL	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL	Elk, Kwiatowa 102	64,2	$\Leftarrow t_2$
...			

- $t_1$ .PESEL =  $t_2$ .PESEL UNKNOWN!!!
- $t_1$ .PESEL =  $t_1$ .Adres UNKNOWN!!!
- $t_1$ .Metraż =  $t_2$ .Metraż

# Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...			
NULL	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL	Elk, Kwiatowa 102	64,2	$\Leftarrow t_2$
...			

- $t_1$ .PESEL =  $t_2$ .PESEL UNKNOWN!!!
- $t_1$ .PESEL =  $t_1$ .Adres UNKNOWN!!!
- $t_1$ .Metraż =  $t_2$ .Metraż TRUE

# Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...			
NULL	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL	Elk, Kwiatowa 102	64,2	$\Leftarrow t_2$
...			

- $t_1$ .PESEL =  $t_2$ .PESEL UNKNOWN!!!
- $t_1$ .PESEL =  $t_1$ .Adres UNKNOWN!!!
- $t_1$ .Metraż =  $t_2$ .Metraż TRUE
- $t_1$ .Adres =  $t_2$ .Adres



# Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...			
NULL	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL	Elk, Kwiatowa 102	64,2	$\Leftarrow t_2$
...			

- $t_1$ .PESEL =  $t_2$ .PESEL UNKNOWN!!!
- $t_1$ .PESEL =  $t_1$ .Adres UNKNOWN!!!
- $t_1$ .Metraż =  $t_2$ .Metraż TRUE
- $t_1$ .Adres =  $t_2$ .Adres FALSE

# Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...			
<b>NULL</b>	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL	Elk, Kwiatowa 102	64,2	$\Leftarrow t_2$
...			

- $t_1$ .PESEL =  $t_2$ .PESEL UNKNOWN!!!
- $t_1$ .PESEL =  $t_1$ .Adres UNKNOWN!!!
- $t_1$ .Metraż =  $t_2$ .Metraż TRUE
- $t_1$ .Adres =  $t_2$ .Adres FALSE
- $t_1$ .PESEL = *NULL*

# Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...			
<b>NULL</b>	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL	Elk, Kwiatowa 102	64,2	$\Leftarrow t_2$
...			

- $t_1$ .PESEL =  $t_2$ .PESEL UNKNOWN!!!
- $t_1$ .PESEL =  $t_1$ .Adres UNKNOWN!!!
- $t_1$ .Metraż =  $t_2$ .Metraż TRUE
- $t_1$ .Adres =  $t_2$ .Adres FALSE
- $t_1$ .PESEL = **NULL** UNKNOWN!!!

# Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...			
<b>NULL</b>	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL	Elk, Kwiatowa 102	64,2	$\Leftarrow t_2$
...			

- $t_1$ .PESEL =  $t_2$ .PESEL UNKNOWN!!!
- $t_1$ .PESEL =  $t_1$ .Adres UNKNOWN!!!
- $t_1$ .Metraż =  $t_2$ .Metraż TRUE
- $t_1$ .Adres =  $t_2$ .Adres FALSE
- $t_1$ .PESEL = **NULL** UNKNOWN!!!
- $t_1$ .PESEL = ''

# Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...			
<b>NULL</b>	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL	Elk, Kwiatowa 102	64,2	$\Leftarrow t_2$
...			

- $t_1$ .PESEL =  $t_2$ .PESEL UNKNOWN!!!
- $t_1$ .PESEL =  $t_1$ .Adres UNKNOWN!!!
- $t_1$ .Metraż =  $t_2$ .Metraż TRUE
- $t_1$ .Adres =  $t_2$ .Adres FALSE
- $t_1$ .PESEL = **NULL** UNKNOWN!!!
- $t_1$ .PESEL = '' UNKNOWN!!!

# Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...			
NULL	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL	Elk, Kwiatowa 102	64,2	$\Leftarrow t_2$
...			

- $t_1$ .PESEL =  $t_2$ .PESEL UNKNOWN!!!
- $t_1$ .PESEL =  $t_1$ .Adres UNKNOWN!!!
- $t_1$ .Metraż =  $t_2$ .Metraż TRUE
- $t_1$ .Adres =  $t_2$ .Adres FALSE
- $t_1$ .PESEL = **NULL** UNKNOWN!!!
- $t_1$ .PESEL = '' UNKNOWN!!!
- IS NULL  $t_1$ .PESEL TRUE
- IS NOT NULL  $t_1$ .Adres TRUE

# Klucze

# Klucze

## Klucz relacji

Podzbiór atrybutów relacji, których wartości zawsze pozwalają jednoznacznie zidentyfikować krotkę relacji. Oznacza, to że nie dopuszczamy, by w danych znalazły się dwie różne krotki o jednakowych wartościach klucza. Relacja może mieć kilka kluczy:

`Student(indeks, PESEL, Nazwisko, ...)`



# Klucze

## Klucz relacji

Podzbiór atrybutów relacji, których wartości zawsze pozwalają jednoznacznie zidentyfikować krotkę relacji. Oznacza, to że nie dopuszczamy, by w danych znalazły się dwie różne krotki o jednakowych wartościach klucza. Relacja może mieć kilka kluczy:

`Student(indeks, PESEL, Nazwisko, ...)`

## Klucz główny

Jeden z kluczy relacji. Zazwyczaj wybieramy ten, według którego najczęściej będziemy wyszukiwać dane z relacji. Pozostałe klucze nazywamy *kandydującymi* lub *alternatywnymi*. Na przykład `indeks` może być kluczem głównym relacji `Student`, a `PESEL` — kluczem alternatywnym.

# Klucze

## Klucz relacji

Podzbiór atrybutów relacji, których wartości zawsze pozwalają jednoznacznie zidentyfikować krotkę relacji. Oznacza, to że nie dopuszczamy, by w danych znalazły się dwie różne krotki o jednakowych wartościach klucza. Relacja może mieć kilka kluczy:

`Student(indeks, PESEL, Nazwisko, ...)`

## Klucz główny

Jeden z kluczy relacji. Zazwyczaj wybieramy ten, według którego najczęściej będziemy wyszukiwać dane z relacji. Pozostałe klucze nazywamy *kandydującymi* lub *alternatywnymi*. Na przykład `indeks` może być kluczem głównym relacji `Student`, a `PESEL` — kluczem alternatywnym.

## Klucz z wielu atrybutów

Stosujemy takie rozwiązanie, gdy jeden atrybut nie wystarcza do zidentyfikowania krotki. Na przykład w relacji `Zaliczenie(indeks, kod_przedmiotu, ocena, data)`.

# Klucz obcy

Dane w bazie muszą często zostać rozmieszczone w różnych relacjach, pomimo że się ze sobą wiążą. Do połączenia danych z różnych relacji służą **klucze obce**.

Zaliczenie:

indeks	kod_przedm	ocena
123456	BD2011	5.0
123456	SK2011	4.5
654321	BD2011	3.5

Student:

indeks	PESEL	nazwisko
123456	AB123456	Abacka
654321	CD345678	Babacka
987654	DE534343	Cabacka

# Klucz obcy

Dane w bazie muszą często zostać rozmieszczone w różnych relacjach, pomimo że się ze sobą wiążą. Do połączenia danych z różnych relacji służą **klucze obce**.

Zaliczenie:

indeks	kod_przedm	ocena
123456	BD2011	5.0
123456	SK2011	4.5
654321	BD2011	3.5

Student:

indeks	PESEL	nazwisko
123456	AB123456	Abacka
654321	CD345678	Babacka
987654	DE534343	Cabacka

- Zamieszczony w relacji Zaliczenie atrybut indeks służy do zidentyfikowania osoby z relacji Student.

# Klucz obcy

Dane w bazie muszą często zostać rozmieszczone w różnych relacjach, pomimo że się ze sobą wiążą. Do połączenia danych z różnych relacji służą **klucze obce**.

Zaliczenie:

indeks	kod_przedm	ocena
123456	BD2011	5.0
123456	SK2011	4.5
654321	BD2011	3.5

Student:

indeks	PESEL	nazwisko
123456	AB123456	Abacka
654321	CD345678	Babacka
987654	DE534343	Cabacka

- Zamieszczony w relacji Zaliczenie atrybut indeks służy do zidentyfikowania osoby z relacji Student.
- W relacji Student atrybut indeks jest kluczem.

# Klucz obcy

Dane w bazie muszą często zostać rozmieszczone w różnych relacjach, pomimo że się ze sobą wiążą. Do połączenia danych z różnych relacji służą **klucze obce**.

Zaliczenie:

indeks	kod_przedm	ocena
123456	BD2011	5.0
123456	SK2011	4.5
654321	BD2011	3.5

Student:

indeks	PESEL	nazwisko
123456	AB123456	Abacka
654321	CD345678	Babacka
987654	DE534343	Cabacka

- Zamieszczony w relacji Zaliczenie atrybut indeks służy do zidentyfikowania osoby z relacji Student.
- W relacji Student atrybut indeks jest kluczem.
- W relacji Zaliczenie atrybut indeks może powtarzać się lub być pusty.

# Klucz obcy

Dane w bazie muszą często zostać rozmieszczone w różnych relacjach, pomimo że się ze sobą wiążą. Do połączenia danych z różnych relacji służą **klucze obce**.

Zaliczenie:

indeks	kod_przedm	ocena
123456	BD2011	5.0
123456	SK2011	4.5
654321	BD2011	3.5

Student:

indeks	PESEL	nazwisko
123456	AB123456	Abacka
654321	CD345678	Babacka
987654	DE534343	Cabacka

- Zamieszczony w relacji `Zaliczenie` atrybut `indeks` służy do zidentyfikowania osoby z relacji `Student`.
- W relacji `Student` atrybut `indeks` jest kluczem.
- W relacji `Zaliczenie` atrybut `indeks` może powtarzać się lub być pusty.
- Jeśli `indeks` jest użyty w relacji `Zaliczenie`, to w relacji `Student` powinna występować osoba o tym indeksie (integralność referencyjna).

# Klucz obcy

Dane w bazie muszą często zostać rozmieszczone w różnych relacjach, pomimo że się ze sobą wiążą. Do połączenia danych z różnych relacji służą **klucze obce**.

Zaliczenie:

indeks	kod_przedm	ocena
123456	BD2011	5.0
123456	SK2011	4.5
654321	BD2011	3.5
999999	BD2012	2.0

Student:

indeks	PESEL	nazwisko
123456	AB123456	Abacka
654321	CD345678	Babacka
987654	DE534343	Cabacka

- Zamieszczony w relacji `Zaliczenie` atrybut `indeks` służy do zidentyfikowania osoby z relacji `Student`.
- W relacji `Student` atrybut `indeks` jest kluczem.
- W relacji `Zaliczenie` atrybut `indeks` może powtarzać się lub być pusty.
- Jeśli `indeks` jest użyty w relacji `Zaliczenie`, to w relacji `Student` powinna występować osoba o tym indeksie (integralność referencyjna).



# Klucz obcy

Dane w bazie muszą często zostać rozmieszczone w różnych relacjach, pomimo że się ze sobą wiążą. Do połączenia danych z różnych relacji służą **klucze obce**.

Zaliczenie:

indeks	kod_przedm	ocena
123456	BD2011	5.0
123456	SK2011	4.5
654321	BD2011	3.5

Student:

indeks	PESEL	nazwisko
123456	AB123456	Abacka
654321	CD345678	Babacka
987654	DE534343	Cabacka

- Zamieszczony w relacji `Zaliczenie` atrybut `indeks` służy do zidentyfikowania osoby z relacji `Student`.
- W relacji `Student` atrybut `indeks` jest kluczem.
- W relacji `Zaliczenie` atrybut `indeks` może powtarzać się lub być pusty.
- Jeśli `indeks` jest użyty w relacji `Zaliczenie`, to w relacji `Student` powinna występować osoba o tym indeksie (integralność referencyjna).

# Więzy — podsumowanie

## Więzy — podsumowanie

**Więzy kolumnowe** — nakładanie ograniczeń na wartość atrybutu: dziedzina, wartość nie pusta (NOT NULL), zakres;

## Więzy — podsumowanie

**Więzy kolumnowe** — nakładanie ograniczeń na wartość atrybutu: dziedzina, wartość nie pusta (NOT NULL), zakres;

**Więzy tabeli** — własność klucza, unikalność w ramach tabeli;

## Więzy — podsumowanie

**Więzy kolumnowe** — nakładanie ograniczeń na wartość atrybutu: dziedzina, wartość nie pusta (NOT NULL), zakres;

**Więzy tabeli** — własność klucza, unikalność w ramach tabeli;

**Więzy między tabelami** — własność klucza obcego;

# Więzy — podsumowanie

**Więzy kolumnowe** — nakładanie ograniczeń na wartość atrybutu: dziedzina, wartość nie pusta (NOT NULL), zakres;

**Więzy tabeli** — własność klucza, unikalność w ramach tabeli;

**Więzy między tabelami** — własność klucza obcego;

**Inne więzy ogólne** — bardziej złożone warunki (np. maksymalnie dwa podejścia do przedmiotu w sesji, dostęp do wybranych przedmiotów dla studentów określonej sekcji, limit liczby osób zapisanych na zajęcia itp.)

# Języki

# Języki

## Język definiowania danych

Musi pozwolić opisać schematy relacji oraz więzy (warunki poprawności) danych.



# Języki

## Język definiowania danych

Musi pozwolić opisać schematy relacji oraz więzy (warunki poprawności) danych.

## Język manipulacji danymi

Pozwala dodawać/usuwać krotki z relacji.

# Języki

## Język definiowania danych

Musi pozwolić opisać schematy relacji oraz więzy (warunki poprawności) danych.

## Język manipulacji danymi

Pozwala dodawać/usuwać krotki z relacji.

## Języki zapytań

Mamy trzy propozycje:

- algebra relacji** — kilka operacji pozwalających działać na relacjach jako na zbiorach;
- relacyjny rachunek dziedzin** — język wykorzystujący formuły logiczne do opisu wartości, które należy znaleźć;
- relacyjny rachunek krotek** — język wykorzystujący formuły logiczne do opisu krotek, które należy znaleźć;

# Języki

## Język definiowania danych

Musi pozwolić opisać schematy relacji oraz więzy (warunki poprawności) danych.

## Język manipulacji danymi

Pozwala dodawać/usuwać krotki z relacji.

## Języki zapytań

Mamy trzy propozycje:

- algebra relacji** — kilka operacji pozwalających działać na relacjach jako na zbiorach;
- relacyjny rachunek dziedzin** — język wykorzystujący formuły logiczne do opisu wartości, które należy znaleźć;
- relacyjny rachunek krotek** — język wykorzystujący formuły logiczne do opisu krotek, które należy znaleźć;

Standard: **SQL**

# Różne podejścia do budowania zapytań

- `selectFrom Student \{ indeks, adres } → do  
pure \{ x: indeks, y: adres }`

# Różne podejścia do budowania zapytań

- `selectFrom Student \{ indeks, adres } → do  
 pure \{ x: indeks, y: adres }`
- `SELECT indeks, adres FROM Student`

# Różne podejścia do budowania zapytań

- `selectFrom Student \{ indeks, adres \} → do  
 pure \{ x: indeks, y: adres \}`
- `SELECT indeks, adres FROM Student`
- `\{(indeks, adres) | ∃ nazwisko Student(indeks, nazwisko, adres)\}`

# Różne podejścia do budowania zapytań

- `selectFrom Student \{ indeks, adres \} → do  
 pure \{ x: indeks, y: adres \}`
- `SELECT indeks, adres FROM Student`
- `\{(indeks, adres) | ∃nazwisko Student(indeks, nazwisko, adres)\}`
- `$\pi_{\{indeks, adres\}}(Student)$`

# Różne podejścia do budowania zapytań

- `selectFrom Student \{ indeks, adres \} → do  
 pure \{ x: indeks, y: adres \}`
- `SELECT indeks, adres FROM Student`
- `\{(indeks, adres) | ∃ nazwisko Student(indeks, nazwisko, adres)\}`
- `π_{indeks, adres}(Student)`
- `for krotka in Student  
 print (krotka.indeks, krotka.adres)`



# Algebra relacji

# Algebra relacji

**Argumentami** są całe relacje (tabele), na których wykonujemy operacje.

# Algebra relacji

**Argumentami** są całe relacje (tabele), na których wykonujemy operacje.

**Zestaw operacji** jest nieliczny: rzutowanie, selekcja, iloczyn kartezjański, suma, różnica i przemianowanie

# Algebra relacji

**Argumentami** są całe relacje (tabele), na których wykonujemy operacje.

**Zestaw operacji** jest nieliczny: rzutowanie, selekcja, iloczyn kartezjański, suma, różnica i przemianowanie

**Zapytanie** to poprawne wyrażenie algebry relacji, a odpowiedź, to wartość tego wyrażenia obliczona na podstawie aktualnego stanu bazy danych.

# Operacje podstawowe - unarne

# Operacje podstawowe - unarne

**Rzut** —  $\pi_{\alpha}(R)$  zwraca relację o schemacie  $\alpha \subseteq attr(R)$  powstałą z obcięcia relacji  $R$  do kolumn  $\alpha$ . Na przykład  $\pi_{nazwisko}(Student)$ .

## Student

Indeks	Nazwisko	Adres
123456	Abacka	Koszalin
654321	Babacka	Szczecin
765678	Cabacka	Koszalin
234565	Abacka	Legnica

# Operacje podstawowe - unarne

**Rzut** —  $\pi_{\alpha}(R)$  zwraca relację o schemacie  $\alpha \subseteq attr(R)$  powstałą z obcięcia relacji  $R$  do kolumn  $\alpha$ . Na przykład  $\pi_{nazwisko}(Student)$ . Duplikaty mogą być eliminowane.

**Student**

Indeks	Nazwisko	Adres
123456	Abacka	Koszalin
654321	Babacka	Szczecin
765678	Cabacka	Koszalin
234565	Abacka	Legnica

**Wynik rzutu na Nazwisko**

Nazwisko
Abacka
Babacka
Cabacka
Abacka

# Operacje podstawowe - unarne

**Rzut** —  $\pi_{\alpha}(R)$  zwraca relację o schemacie  $\alpha \subseteq attr(R)$  powstałą z obcięcia relacji  $R$  do kolumn  $\alpha$ . Na przykład  $\pi_{nazwisko}(Student)$ . Duplikaty mogą być eliminowane.

**Selekcja** —  $\sigma_F(R)$  zwraca krotki wybrane z relacji  $R$  spełniające warunek  $F$ . Na przykład  $\sigma_{Adres='Koszalin'}(Student)$ .

## Student

Indeks	Nazwisko	Adres
123456	Abacka	Koszalin
654321	Babacka	Szczecin
765678	Cabacka	Koszalin
234565	Abacka	Legnica



# Operacje podstawowe - unarne

**Rzut** —  $\pi_{\alpha}(R)$  zwraca relację o schemacie  $\alpha \subseteq attr(R)$  powstałą z obcięcia relacji  $R$  do kolumn  $\alpha$ . Na przykład  $\pi_{nazwisko}(Student)$ . Duplikaty mogą być eliminowane.

**Selekcja** —  $\sigma_F(R)$  zwraca krotki wybrane z relacji  $R$  spełniające warunek  $F$ . Na przykład  $\sigma_{Adres='Koszalin'}(Student)$ .

## Student

Indeks	Nazwisko	Adres
123456	Abacka	Koszalin
654321	Babacka	Szczecin
765678	Cabacka	Koszalin
234565	Abacka	Legnica

## Wynik selekcji Adres='Koszalin'

Indeks	Nazwisko	Adres
123456	Abacka	Koszalin
765678	Cabacka	Koszalin

# Operacje podstawowe - unarne

**Rzut** —  $\pi_{\alpha}(R)$  zwraca relację o schemacie  $\alpha \subseteq attr(R)$  powstałą z obcięcia relacji  $R$  do kolumn  $\alpha$ . Na przykład  $\pi_{nazwisko}(Student)$ . Duplikaty mogą być eliminowane.

**Selekcja** —  $\sigma_F(R)$  zwraca krotki wybrane z relacji  $R$  spełniające warunek  $F$ . Na przykład  $\sigma_{Adres='Koszalin'}(Student)$ .

**Przemianowanie** —  $\rho_{S(B_1, \dots, B_k)}(R)$  zmienia nazwę relacji  $R$  na  $S$  i nazwy odpowiednich atrybutów  $R$  na  $B_1, \dots, B_k$ . Na przykład  
 $\rho_{Osoba(id, nazwisko, miasto)}(\pi_{indeks, nazwisko, adres}(Student))$ .

## Student

Indeks	Nazwisko	Adres
123456	Abacka	Koszalin
654321	Babacka	Szczecin
765678	Cabacka	Koszalin
234565	Abacka	Legnica

# Operacje podstawowe - unarne

**Rzut** —  $\pi_{\alpha}(R)$  zwraca relację o schemacie  $\alpha \subseteq \text{attr}(R)$  powstałą z obcięcia relacji  $R$  do kolumn  $\alpha$ . Na przykład  $\pi_{\text{nazwisko}}(\text{Student})$ . Duplikaty mogą być eliminowane.

**Selekcja** —  $\sigma_F(R)$  zwraca krotki wybrane z relacji  $R$  spełniające warunek  $F$ . Na przykład  $\sigma_{\text{Adres}='Koszalin'}(\text{Student})$ .

**Przemianowanie** —  $\rho_{S(B_1, \dots, B_k)}(R)$  zmienia nazwę relacji  $R$  na  $S$  i nazwy odpowiednich atrybutów  $R$  na  $B_1, \dots, B_k$ . Na przykład  $\rho_{\text{Osoba}(\text{id}, \text{nazwisko}, \text{miasto})}(\pi_{\text{indeks}, \text{nazwisko}, \text{adres}}(\text{Student}))$ .

## Student

Indeks	Nazwisko	Adres
123456	Abacka	Koszalin
654321	Babacka	Szczecin
765678	Cabacka	Koszalin
234565	Abacka	Legnica

## Tabela po przemianowaniu: Osoba

Id	Nazwisko	Miasto
123456	Abacka	Koszalin
654321	Babacka	Szczecin
765678	Cabacka	Koszalin
234565	Abacka	Legnica

# Operacje teoriomnogościowe — suma, różnica, przekrój

Suma ( $\cup$ ), różnica ( $\setminus$ ), przekrój ( $\cap$ ) — „zwykłe” operacje na zbiorach;  $R \setminus S$  i  $R \cup S$  wymagają, by  $attr(R) = attr(S)$ ; w praktyce mógł być zastępowane operacjami na **wielozbiorach**. Dodawane (odejmowane, krojone) relacje muszą mieć zgodne schematy.

**StudentII**

Indeks	Nazwisko	Adres
123456	Abacka	Koszalin
654321	Babacka	Szczecin
234565	Abacka	Legnica

**StudentIM**

Indeks	Nazwisko	Adres
012345	Zetowski	Kielce
654321	Babacka	Szczecin

**Relacja wynikowa:**

# Operacje teoriomnogościowe — suma, różnica, przekrój

Suma ( $\cup$ ), różnica ( $\setminus$ ), przekrój ( $\cap$ ) — „zwykłe” operacje na zbiorach;  $R \setminus S$  i  $R \cup S$  wymagają, by  $attr(R) = attr(S)$ ; w praktyce mogą być zastępowane operacjami na **wielozbiorach**. Dodawane (odejmowane, krojone) relacje muszą mieć zgodne schematy.

**StudentII**

Indeks	Nazwisko	Adres
123456	Abacka	Koszalin
654321	Babacka	Szczecin
234565	Abacka	Legnica

**StudentIM**

Indeks	Nazwisko	Adres
012345	Zetowski	Kielce
654321	Babacka	Szczecin

**Relacja wynikowa:**

Indeks	Nazwisko	Adres

# Operacje teoriomnogościowe — suma, różnica, przekrój

**Suma ( $\cup$ ), różnica ( $\setminus$ ), przekrój ( $\cap$ )** — „zwykłe” operacje na zbiorach;  $R \setminus S$  i  $R \cup S$  wymagają, by  $attr(R) = attr(S)$ ; w praktyce mogą być zastępowane operacjami na **wielozbiorach**. Dodawane (odejmowane, krojone) relacje muszą mieć zgodne schematy.

**StudentII**

Indeks	Nazwisko	Adres
123456	Abacka	Koszalin
654321	Babacka	Szczecin
234565	Abacka	Legnica

**StudentIM**

Indeks	Nazwisko	Adres
012345	Zetowski	Kielce
654321	Babacka	Szczecin

**Relacja wynikowa:**

Indeks	Nazwisko	Adres

# Operacje teoriomnogościowe — suma, różnica, przekrój

**Suma ( $\cup$ ), różnica ( $\setminus$ ), przekrój ( $\cap$ )** — „zwykłe” operacje na zbiorach;  $R \setminus S$  i  $R \cup S$  wymagają, by  $attr(R) = attr(S)$ ; w praktyce mogą być zastępowane operacjami na **wielozbiorach**. Dodawane (odejmowane, krojone) relacje muszą mieć zgodne schematy.

**StudentII**

Indeks	Nazwisko	Adres
123456	Abacka	Koszalin
654321	Babacka	Szczecin
234565	Abacka	Legnica

**StudentIM**

Indeks	Nazwisko	Adres
012345	Zetowski	Kielce
654321	Babacka	Szczecin

**Relacja wynikowa:**

Indeks	Nazwisko	Adres

# Operacje teoriomnogościowe — suma, różnica, przekrój

**Suma ( $\cup$ ), różnica ( $\setminus$ ), przekrój ( $\cap$ )** — „zwykłe” operacje na zbiorach;  $R \setminus S$  i  $R \cup S$  wymagają, by  $attr(R) = attr(S)$ ; w praktyce mogą być zastępowane operacjami na **wielozbiorach**. Dodawane (odejmowane, krojone) relacje muszą mieć zgodne schematy.

**StudentII**

Indeks	Nazwisko	Adres
123456	Abacka	Koszalin
654321	Babacka	Szczecin
234565	Abacka	Legnica

**StudentIM**

Indeks	Nazwisko	Adres
012345	Zetowski	Kielce
654321	Babacka	Szczecin

**Relacja wynikowa:**

Indeks	Nazwisko	Adres
123456	Abacka	Koszalin
654321	Babacka	Szczecin
234565	Abacka	Legnica
012345	Zetowski	Kielce
654321	Babacka	Szczecin



# Operacje teoriomnogościowe — suma, różnica, przekrój

Suma ( $\cup$ ), różnica ( $\setminus$ ), przekrój ( $\cap$ ) — „zwykłe” operacje na zbiorach;  $R \setminus S$  i  $R \cup S$  wymagają, by  $attr(R) = attr(S)$ ; w praktyce mogą być zastępowane operacjami na **wielozbiorach**. Dodawane (odejmowane, krojone) relacje muszą mieć zgodne schematy.

**StudentII**

Indeks	Nazwisko	Adres
123456	Abacka	Koszalin
654321	Babacka	Szczecin
234565	Abacka	Legnica

**StudentIM**

Indeks	Nazwisko	Adres
012345	Zetowski	Kielce
654321	Babacka	Szczecin

**Relacja wynikowa:**

Indeks	Nazwisko	Adres

# Operacje teoriomnogościowe — suma, różnica, przekrój

Suma ( $\cup$ ), różnica ( $\setminus$ ), przekrój ( $\cap$ ) — „zwykłe” operacje na zbiorach;  $R \setminus S$  i  $R \cup S$  wymagają, by  $attr(R) = attr(S)$ ; w praktyce mogą być zastępowane operacjami na **wielozbiorach**. Dodawane (odejmowane, krojone) relacje muszą mieć zgodne schematy.

StudentII

Indeks	Nazwisko	Adres
123456	Abacka	Koszalin
654321	Babacka	Szczecin
234565	Abacka	Legnica

StudentIM

Indeks	Nazwisko	Adres
012345	Zetowski	Kielce
654321	Babacka	Szczecin

Relacja wynikowa:

Indeks	Nazwisko	Adres

# Operacje teoriomnogościowe — suma, różnica, przekrój

Suma ( $\cup$ ), różnica ( $\setminus$ ), przekrój ( $\cap$ ) — „zwykłe” operacje na zbiorach;  $R \setminus S$  i  $R \cup S$  wymagają, by  $attr(R) = attr(S)$ ; w praktyce mógł być zastępowane operacjami na **wielozbiorach**. Dodawane (odejmowane, krojone) relacje muszą mieć zgodne schematy.

StudentII

Indeks	Nazwisko	Adres
123456	Abacka	Koszalin
654321	Babacka	Szczecin
234565	Abacka	Legnica

StudentIM

Indeks	Nazwisko	Adres
012345	Zetowski	Kielce
654321	Babacka	Szczecin

Relacja wynikowa:

Indeks	Nazwisko	Adres
123456	Abacka	Koszalin
234565	Abacka	Legnica

# Operacje teoriomnogościowe — suma, różnica, przekrój

Suma ( $\cup$ ), różnica ( $\setminus$ ), **przekrój** ( $\cap$ ) — „zwykłe” operacje na zbiorach;  $R \setminus S$  i  $R \cup S$  wymagają, by  $attr(R) = attr(S)$ ; w praktyce mogły być zastępowane operacjami na **wielozbiorach**. Dodawane (odejmowane, krojone) relacje muszą mieć zgodne schematy.

**StudentII**

Indeks	Nazwisko	Adres
123456	Abacka	Koszalin
654321	Babacka	Szczecin
234565	Abacka	Legnica

**StudentIM**

Indeks	Nazwisko	Adres
012345	Zetowski	Kielce
654321	Babacka	Szczecin

**Relacja wynikowa:**

Indeks	Nazwisko	Adres

# Operacje teoriomnogościowe — suma, różnica, przekrój

Suma ( $\cup$ ), różnica ( $\setminus$ ), **przekrój** ( $\cap$ ) — „zwykłe” operacje na zbiorach;  $R \setminus S$  i  $R \cup S$  wymagają, by  $attr(R) = attr(S)$ ; w praktyce mógł być zastępowane operacjami na **wielozbiorach**. Dodawane (odejmowane, krojone) relacje muszą mieć zgodne schematy.

StudentII

Indeks	Nazwisko	Adres
123456	Abacka	Koszalin
<b>654321</b>	<b>Babacka</b>	<b>Szczecin</b>
234565	Abacka	Legnica

StudentIM

Indeks	Nazwisko	Adres
012345	Zetowski	Kielce
<b>654321</b>	<b>Babacka</b>	<b>Szczecin</b>

Relacja wynikowa:

Indeks	Nazwisko	Adres

# Operacje teoriomnogościowe — suma, różnica, przekrój

Suma ( $\cup$ ), różnica ( $\setminus$ ), **przekrój** ( $\cap$ ) — „zwykłe” operacje na zbiorach;  $R \setminus S$  i  $R \cup S$  wymagają, by  $attr(R) = attr(S)$ ; w praktyce mógł być zastępowane operacjami na **wielozbiorach**. Dodawane (odejmowane, krojone) relacje muszą mieć zgodne schematy.

StudentII

Indeks	Nazwisko	Adres
123456	Abacka	Koszalin
<b>654321</b>	<b>Babacka</b>	<b>Szczecin</b>
234565	Abacka	Legnica

StudentIM

Indeks	Nazwisko	Adres
012345	Zetowski	Kielce
<b>654321</b>	<b>Babacka</b>	<b>Szczecin</b>

Relacja wynikowa:

Indeks	Nazwisko	Adres
654321	Babacka	Szczecin

# Złączenia

**Iloczyn kartezjański ( $\times$ )** — dla relacji o rozłącznych schematach ( $attr(R) \cap attr(S) = \emptyset$ )  $R \times S$  jest relacją o atrybutach  $attr(R) \cup attr(S)$  zawierającą krotki  $t = rs$ , gdzie  $r \in R$  i  $s \in S$  oraz  $t.attr(R) = r$  i  $t.attr(S) = s$ .

# Złączenia

**Iloczyn kartezjański** ( $\times$ ) — dla relacji o rozłącznych schematach ( $attr(R) \cap attr(S) = \emptyset$ )  $R \times S$  jest relacją o atrybutach  $attr(R) \cup attr(S)$  zawierającą krotki  $t = rs$ , gdzie  $r \in R$  i  $s \in S$  oraz  $t.attr(R) = r$  i  $t.attr(S) = s$ .

## Student

Indeks	Nazwisko	Adres
123456	Abacka	Koszalin
654321	Babacka	Szczecin
234565	Abacka	Legnica

## Przedmiot

Kod	Nazwa	Typ
BD	Bazy danych	podst
AM	Analiza mat.	obow



# Złączenia

**Iloczyn kartezjański ( $\times$ )** — dla relacji o rozłącznych schematach ( $attr(R) \cap attr(S) = \emptyset$ )  $R \times S$  jest relacją o atrybutach  $attr(R) \cup attr(S)$  zawierająca krotki  $t = rs$ , gdzie  $r \in R$  i  $s \in S$  oraz  $t.attr(R) = r$  i  $t.attr(S) = s$ .

## Student

Indeks	Nazwisko	Adres
123456	Abacka	Koszalin
654321	Babacka	Szczecin
234565	Abacka	Legnica

## Przedmiot

Kod	Nazwa	Typ
BD	Bazy danych	podst
AM	Analiza mat.	obow

## Student $\times$ Przedmiot

Indeks	Nazwisko	Adres	Kod	Nazwa	Typ
123456	Abacka	Koszalin	BD	Bazy danych	podst
654321	Babacka	Szczecin	BD	Bazy danych	podst
234565	Abacka	Legnica	BD	Bazy danych	podst
123456	Abacka	Koszalin	AM	Analiza mat.	obow
654321	Babacka	Szczecin	AM	Analiza mat.	obow
234565	Abacka	Legnica	AM	Analiza mat.	obow

# Złączenia

**Iloczyn kartezjański ( $\times$ )** — dla relacji o rozłącznych schematach ( $attr(R) \cap attr(S) = \emptyset$ )  $R \times S$  jest relacją o atrybutach  $attr(R) \cup attr(S)$  zawierająca krotki  $t = rs$ , gdzie  $r \in R$  i  $s \in S$  oraz  $t.attr(R) = r$  i  $t.attr(S) = s$ .

## Student

Indeks	Nazwisko	Adres
123456	Abacka	Koszalin
654321	Babacka	Szczecin
234565	Abacka	Legnica

## Przedmiot

Kod	Nazwa	Typ
BD	Bazy danych	podst
AM	Analiza mat.	obow

## Student $\times$ Przedmiot

Indeks	Nazwisko	Adres	Kod	Nazwa	Typ
123456	Abacka	Koszalin	BD	Bazy danych	podst
654321	Babacka	Szczecin	BD	Bazy danych	podst
234565	Abacka	Legnica	BD	Bazy danych	podst
123456	Abacka	Koszalin	AM	Analiza mat.	obow
654321	Babacka	Szczecin	AM	Analiza mat.	obow
234565	Abacka	Legnica	AM	Analiza mat.	obow

# Złączenia

**Iloczyn kartezjański ( $\times$ )** — dla relacji o rozłącznych schematach ( $attr(R) \cap attr(S) = \emptyset$ )  $R \times S$  jest relacją o atrybutach  $attr(R) \cup attr(S)$  zawierająca krotki  $t = rs$ , gdzie  $r \in R$  i  $s \in S$  oraz  $t.attr(R) = r$  i  $t.attr(S) = s$ .

## Student

Indeks	Nazwisko	Adres
123456	Abacka	Koszalin
654321	Babacka	Szczecin
234565	Abacka	Legnica

## Przedmiot

Kod	Nazwa	Typ
BD	Bazy danych	podst
AM	Analiza mat.	obow

## Student $\times$ Przedmiot

Indeks	Nazwisko	Adres	Kod	Nazwa	Typ
123456	Abacka	Koszalin	BD	Bazy danych	podst
654321	Babacka	Szczecin	BD	Bazy danych	podst
234565	Abacka	Legnica	BD	Bazy danych	podst
123456	Abacka	Koszalin	AM	Analiza mat.	obow
654321	Babacka	Szczecin	AM	Analiza mat.	obow
234565	Abacka	Legnica	AM	Analiza mat.	obow

# Złączenie naturalne

**Złączenie naturalne ( $\bowtie$ )** Dla relacji  $R$  i  $S$  *złączeniem naturalnym*  $R \bowtie S$  jest relacja o schemacie  $attr(R) \cup attr(S)$  zawierająca krotki  $t$ , dla których istnieją krotki  $r \in R$  i  $s \in S$ , takie że  $r.(attr(R) \cap attr(S)) = s.(attr(R) \cap attr(S))$  oraz  $t.attr(R) = r$  i  $t.attr(S) = s$ .

## Student

Indeks	Nazwisko	Adres
654321	Babacka	Szczecin
234565	Abacka	Legnica
123456	Abacka	Koszalin

## Ocena

Indeks	Kod	Stopien
654321	BD	5.0
234565	BD	4.5
234565	AM	4.5
012345	AM	3.5

# Złączenie naturalne

**Złączenie naturalne ( $\bowtie$ )** Dla relacji  $R$  i  $S$  *złączeniem naturalnym*  $R \bowtie S$  jest relacja o schemacie  $attr(R) \cup attr(S)$  zawierająca krotki  $t$ , dla których istnieją krotki  $r \in R$  i  $s \in S$ , takie że  $r.(attr(R) \cap attr(S)) = s.(attr(R) \cap attr(S))$  oraz  $t.attr(R) = r$  i  $t.attr(S) = s$ .

## Student

Indeks	Nazwisko	Adres
654321	Babacka	Szczecin
234565	Abacka	Legnica
123456	Abacka	Koszalin

## Ocena

Indeks	Kod	Stopien
654321	BD	5.0
234565	BD	4.5
234565	AM	4.5
012345	AM	3.5

# Złączenie naturalne

**Złączenie naturalne ( $\bowtie$ )** Dla relacji  $R$  i  $S$  *złączeniem naturalnym*  $R \bowtie S$  jest relacja o schemacie  $attr(R) \cup attr(S)$  zawierająca krotki  $t$ , dla których istnieją krotki  $r \in R$  i  $s \in S$ , takie że  $r.(attr(R) \cap attr(S)) = s.(attr(R) \cap attr(S))$  oraz  $t.attr(R) = r$  i  $t.attr(S) = s$ .

## Student

Indeks	Nazwisko	Adres
654321	Babacka	Szczecin
234565	Abacka	Legnica
123456	Abacka	Koszalin

## Ocena

Indeks	Kod	Stopien
654321	BD	5.0
234565	BD	4.5
234565	AM	4.5
012345	AM	3.5

## Student $\bowtie$ Ocena

Indeks	Nazwisko	Adres	Kod	Stopien

# Złączenie naturalne

**Złączenie naturalne ( $\bowtie$ )** Dla relacji  $R$  i  $S$  *złączeniem naturalnym*  $R \bowtie S$  jest relacja o schemacie  $attr(R) \cup attr(S)$  zawierająca krotki  $t$ , dla których istnieją krotki  $r \in R$  i  $s \in S$ , takie że  $r.(attr(R) \cap attr(S)) = s.(attr(R) \cap attr(S))$  oraz  $t.attr(R) = r$  i  $t.attr(S) = s$ .

## Student

Indeks	Nazwisko	Adres
654321	Babacka	Szczecin
234565	Abacka	Legnica
123456	Abacka	Koszalin

## Ocena

Indeks	Kod	Stopien
654321	BD	5.0
234565	BD	4.5
234565	AM	4.5
012345	AM	3.5

## Student $\bowtie$ Ocena

Indeks	Nazwisko	Adres	Kod	Stopien

# Złączenie naturalne

**Złączenie naturalne ( $\bowtie$ )** Dla relacji  $R$  i  $S$  *złączeniem naturalnym*  $R \bowtie S$  jest relacja o schemacie  $attr(R) \cup attr(S)$  zawierająca krotki  $t$ , dla których istnieją krotki  $r \in R$  i  $s \in S$ , takie że  $r.(attr(R) \cap attr(S)) = s.(attr(R) \cap attr(S))$  oraz  $t.attr(R) = r$  i  $t.attr(S) = s$ .

**Student**

Indeks	Nazwisko	Adres
654321	Babacka	Szczecin
234565	Abacka	Legnica
123456	Abacka	Koszalin

**Ocena**

Indeks	Kod	Stopien
654321	BD	5.0
234565	BD	4.5
234565	AM	4.5
012345	AM	3.5

**Student  $\bowtie$  Ocena**

Indeks	Nazwisko	Adres	Kod	Stopien
654321	Babacka	Szczecin	BD	5.0



# Złączenie naturalne

**Złączenie naturalne ( $\bowtie$ )** Dla relacji  $R$  i  $S$  *złączeniem naturalnym*  $R \bowtie S$  jest relacja o schemacie  $attr(R) \cup attr(S)$  zawierająca krotki  $t$ , dla których istnieją krotki  $r \in R$  i  $s \in S$ , takie że  $r.(attr(R) \cap attr(S)) = s.(attr(R) \cap attr(S))$  oraz  $t.attr(R) = r$  i  $t.attr(S) = s$ .

## Student

Indeks	Nazwisko	Adres
654321	Babacka	Szczecin
234565	Abacka	Legnica
123456	Abacka	Koszalin

## Ocena

Indeks	Kod	Stopien
654321	BD	5.0
234565	BD	4.5
234565	AM	4.5
012345	AM	3.5

## Student $\bowtie$ Ocena

Indeks	Nazwisko	Adres	Kod	Stopien
654321	Babacka	Szczecin	BD	5.0
234565	Abacka	Legnica	BD	4.5
234565	Abacka	Legnica	AM	3.5

# Złączenie naturalne

**Złączenie naturalne ( $\bowtie$ )** Dla relacji  $R$  i  $S$  *złączeniem naturalnym*  $R \bowtie S$  jest relacja o schemacie  $attr(R) \cup attr(S)$  zawierająca krotki  $t$ , dla których istnieją krotki  $r \in R$  i  $s \in S$ , takie że  $r.(attr(R) \cap attr(S)) = s.(attr(R) \cap attr(S))$  oraz  $t.attr(R) = r$  i  $t.attr(S) = s$ .

## Student

Indeks	Nazwisko	Adres
654321	Babacka	Szczecin
234565	Abacka	Legnica
123456	Abacka	Koszalin

## Ocena

Indeks	Kod	Stopien
654321	BD	5.0
234565	BD	4.5
234565	AM	4.5
012345	AM	3.5

## Student $\bowtie$ Ocena

Indeks	Nazwisko	Adres	Kod	Stopien
654321	Babacka	Szczecin	BD	5.0
234565	Abacka	Legnica	BD	4.5
234565	Abacka	Legnica	AM	3.5

Krotki, które nie mają pary, nie wchodzi do wyniku!

# Wszystkie operacje algebry relacji

# Wszystkie operacje algebry relacji

Złączenie  $\theta_F$  to iloczyn kartezjański połączony z selekcją:

$$R \bowtie_F S = \sigma_F(R \times S)$$

# Wszystkie operacje algebry relacji

Złączenie  $\theta_F$  to iloczyn kartezjański połączony z selekcją:

$$R \bowtie_F S = \sigma_F(R \times S)$$

Złączenia zewnętrzne to złączenie naturalne, do którego wyniku dorzuca się krotki, które nie znalazły pary. W polach, które są niewypełnione, wpisywana jest wartość NULL.

# Wszystkie operacje algebry relacji

Złączenie  $\theta_F$  to iloczyn kartezjański połączony z selekcją:

$$R \bowtie_F S = \sigma_F(R \times S)$$

**Złączenia zewnętrzne** to złączenie naturalne, do którego wyniku dorzuca się krotki, które nie znalazły pary. W polach, które są niewypełnione, wpisywana jest wartość NULL.

**Półzłączenia** to operacja wybierająca z relacji krotki, które połączyłyby się, gdyby wykonywano złączenie naturalne.

# Wszystkie operacje algebry relacji

Złączenie  $\theta_F$  to iloczyn kartezjański połączony z selekcją:

$$R \bowtie_F S = \sigma_F(R \times S)$$

**Złączenia zewnętrzne** to złączenie naturalne, do którego wyniku dorzuca się krotki, które nie znalazły pary. W polach, które są niewypełnione, wpisywana jest wartość NULL.

**Półzłączenia** to operacja wybierająca z relacji krotki, które połączyłyby się, gdyby wykonywano złączenie naturalne.

**Inne operacje** np. iloraz, złączenie lewostronne i prawostronne.

# Wszystkie operacje algebry relacji

Złączenie  $\theta_F$  to iloczyn kartezjański połączony z selekcją:

$$R \bowtie_F S = \sigma_F(R \times S)$$

**Złączenia zewnętrzne** to złączenie naturalne, do którego wyniku dorzuca się krotki, które nie znalazły pary. W polach, które są niewypełnione, wpisywana jest wartość NULL.

**Półzłączenia** to operacja wybierająca z relacji krotki, które połączyłyby się, gdyby wykonywano złączenie naturalne.

**Inne operacje** np. iloraz, złączenie lewostronne i prawostronne.

**Zapytania** budujemy poprawne wyrażenia używając operatorów algebry relacji, nawiasów i stałych.



# Wszystkie operacje algebry relacji

Złączenie  $\theta_F$  to iloczyn kartezjański połączony z selekcją:

$$R \bowtie_F S = \sigma_F(R \times S)$$

**Złączenia zewnętrzne** to złączenie naturalne, do którego wyniku dorzuca się krotki, które nie znalazły pary. W polach, które są niewypełnione, wpisywana jest wartość NULL.

**Półzłączenia** to operacja wybierająca z relacji krotki, które połączyłyby się, gdyby wykonywano złączenie naturalne.

**Inne operacje** np. iloraz, złączenie lewostronne i prawostronne.

**Zapytania** budujemy poprawne wyrażenia używając operatorów algebry relacji, nawiasów i stałych.

Wszystkie operacje algebry relacji są wyrażalne za pomocą:  $\pi$ ,  $\sigma$ ,  $\rho$ ,  $\times$ ,  $\cup$ ,  $\setminus$ .

# Baza do przykładów

- **Student**=(indeks,nazwisko, rok), czyli indeks, nazwisko i rok studiów studenta;
- **Przedmiot**=(nazwa, typ), czyli nazwa i typ przedmiotu;
- **Ocena**=(indeks,przed,data,stop), czyli ocena uzyskana przez studenta za przedmiot wraz z datą wystawienia.

Klucze główne relacji są podkreślone. Dodatkowo w relacji *O* występują klucze obce:

- *O.indeks* odnoszący się do *S.indeks*,
- *O.przed* odnoszący się do *P.nazwa*,
- Czy pola *data* i *stop* w relacji *Ocena* mogą być puste?

# Przykłady 1-3

## Baza danych

$S = (\underline{\text{indeks}}, \text{nazwisko}, \text{rok}), P = (\underline{\text{nazwa}}, \text{typ}), O = (\underline{\text{indeks}}, \underline{\text{przed}}, \underline{\text{data}}, \text{stop})$

# Przykłady 1-3

## Baza danych

$S = (\underline{indeks}, nazwisko, rok), P = (\underline{nazwa}, typ), O = (\underline{indeks}, \underline{przed}, \underline{data}, stop)$

1.  $\pi_{S.indeks, nazwisko}(\sigma_{stop=5.0 \wedge przed='BD'}(S \bowtie O));$

## Znaczenie zapytań

# Przykłady 1-3

## Baza danych

$S = (\underline{\text{indeks}}, \text{nazwisko}, \text{rok}), P = (\underline{\text{nazwa}}, \text{typ}), O = (\underline{\text{indeks}}, \underline{\text{przed}}, \underline{\text{data}}, \text{stop})$

1.  $\pi_{S.\text{indeks}, \text{nazwisko}}(\sigma_{\text{stop}=5.0 \wedge \text{przed}='BD'}(S \bowtie O));$

## Znaczenie zapytań

1. Indeksy i nazwiska studentów, którzy dostali 5.0 z BD.

# Przykłady 1-3

## Baza danych

$S = (\underline{\text{indeks}}, \text{nazwisko}, \text{rok}), P = (\underline{\text{nazwa}}, \text{typ}), O = (\underline{\text{indeks}}, \underline{\text{przed}}, \underline{\text{data}}, \text{stop})$

1.  $\pi_{S.\text{indeks}, \text{nazwisko}}(\sigma_{\text{stop}=5.0 \wedge \text{przed}='BD'}(S \bowtie O));$

## Znaczenie zapytań

1. Indeksy i nazwiska studentów, którzy dostali 5.0 z BD.
2. Pełne dane studentów, którzy dostali jakąś ocenę 5.0.

# Przykłady 1-3

## Baza danych

$S = (\underline{\text{indeks}}, \text{nazwisko}, \text{rok}), P = (\underline{\text{nazwa}}, \text{typ}), O = (\underline{\text{indeks}}, \underline{\text{przed}}, \underline{\text{data}}, \text{stop})$

1.  $\pi_{S.\text{indeks}, \text{nazwisko}}(\sigma_{\text{stop}=5.0 \wedge \text{przed}='BD'}(S \bowtie O));$
2.  $\pi_{S.\text{indeks}, \text{nazwisko}, \text{rok}}(S \bowtie \sigma_{\text{stop}=5.0}(O));$

## Znaczenie zapytań

1. Indeksy i nazwiska studentów, którzy dostali 5.0 z BD.
2. Pełne dane studentów, którzy dostali jakąś ocenę 5.0.

# Przykłady 1-3

## Baza danych

$S = (\underline{\text{indeks}}, \text{nazwisko}, \text{rok}), P = (\underline{\text{nazwa}}, \text{typ}), O = (\underline{\text{indeks}}, \underline{\text{przed}}, \underline{\text{data}}, \text{stop})$

1.  $\pi_{S.\text{indeks}, \text{nazwisko}}(\sigma_{\text{stop}=5.0 \wedge \text{przed}='BD'}(S \bowtie O));$
2.  $\pi_{S.\text{indeks}, \text{nazwisko}, \text{rok}}(S \bowtie \sigma_{\text{stop}=5.0}(O));$

## Znaczenie zapytań

1. Indeksy i nazwiska studentów, którzy dostali 5.0 z BD.
2. Pełne dane studentów, którzy dostali jakąś ocenę 5.0.
3. Studenci, którzy podchodzili do BD co najmniej dwa razy.



# Przykłady 1-3

## Baza danych

$S = (\underline{\text{indeks}}, \text{nazwisko}, \text{rok}), P = (\underline{\text{nazwa}}, \text{typ}), O = (\underline{\text{indeks}}, \underline{\text{przed}}, \underline{\text{data}}, \text{stop})$

1.  $\pi_{S.\text{indeks}, \text{nazwisko}}(\sigma_{\text{stop}=5.0 \wedge \text{przed}='BD'}(S \bowtie O));$
2.  $\pi_{S.\text{indeks}, \text{nazwisko}, \text{rok}}(S \bowtie \sigma_{\text{stop}=5.0}(O));$
3.  $\pi_{S.\text{indeks}, \text{nazwisko}}(S \bowtie \sigma_{i1=\text{indeks} \wedge p1=\text{przed} \wedge \text{przed}='BD' \wedge \text{data} \neq d1}(\rho_{O1(i1, p1, d1, s1)}(O) \times O)).$

## Znaczenie zapytań

1. Indeksy i nazwiska studentów, którzy dostali 5.0 z BD.
2. Pełne dane studentów, którzy dostali jakąś ocenę 5.0.
3. Studenci, którzy podchodzili do BD co najmniej dwa razy.

## Przykład 4

### Baza danych

$S = (\underline{\text{indeks}}, \text{nazwisko}, \text{rok}), P = (\underline{\text{nazwa}}, \text{typ}), O = (\underline{\text{indeks}}, \underline{\text{przed}}, \underline{\text{data}}, \text{stop})$

Studenci, którzy nie dostali 5.0.

## Przykład 4

### Baza danych

$S = (\underline{indeks}, nazwisko, rok)$ ,  $P = (\underline{nazwa}, typ)$ ,  $O = (\underline{indeks}, \underline{przed}, \underline{data}, stop)$

Studenci, którzy nie dostali 5.0.

4a.  $\pi_{S.indeks, nazwisko, rok}(S \bowtie \sigma_{stop \neq 5.0}(O));$

### Znaczenie zapytań

## Przykład 4

### Baza danych

$S = (\underline{indeks}, nazwisko, rok)$ ,  $P = (\underline{nazwa}, typ)$ ,  $O = (\underline{indeks}, \underline{przed}, \underline{data}, stop)$

Studenci, którzy nie dostali 5.0.

4a.  $\pi_{S.indeks, nazwisko, rok}(S \bowtie \sigma_{stop \neq 5.0}(O));$

### Znaczenie zapytań

4a. Studenci, którzy dostali jakąś ocenę inną niż 5.0.

## Przykład 4

### Baza danych

$S = (\underline{indeks}, nazwisko, rok)$ ,  $P = (\underline{nazwa}, typ)$ ,  $O = (\underline{indeks}, \underline{przed}, \underline{data}, stop)$

Studenci, którzy nie dostali 5.0.

4a.  $\pi_{S.indeks, nazwisko, rok}(S \bowtie \sigma_{stop \neq 5.0}(O));$

4b.  $\pi_{S.indeks, nazwisko, rok}(S \bowtie \sigma_{stop \text{ IS NULL }}(O));$

### Znaczenie zapytań

4a. Studenci, którzy dostali jakąś ocenę inną niż 5.0.

## Przykład 4

### Baza danych

$S = (\underline{indeks}, nazwisko, rok)$ ,  $P = (\underline{nazwa}, typ)$ ,  $O = (\underline{indeks}, \underline{przed}, \underline{data}, stop)$

Studenci, którzy nie dostali 5.0.

4a.  $\pi_{S.indeks, nazwisko, rok}(S \bowtie \sigma_{stop \neq 5.0}(O));$

4b.  $\pi_{S.indeks, nazwisko, rok}(S \bowtie \sigma_{stop \text{ IS NULL }}(O));$

### Znaczenie zapytań

4a. Studenci, którzy dostali jakąś ocenę inną niż 5.0.

4b. Studenci, którzy nie dostali wpisu (niezgodne z więzami relacji)

## Przykład 4

### Baza danych

$S = (\underline{indeks}, nazwisko, rok)$ ,  $P = (\underline{nazwa}, typ)$ ,  $O = (\underline{indeks}, \underline{przed}, \underline{data}, stop)$

Studenci, którzy nie dostali 5.0.

- 4a.  $\pi_{S.indeks, nazwisko, rok}(S \bowtie \sigma_{stop \neq 5.0}(O));$
- 4b.  $\pi_{S.indeks, nazwisko, rok}(S \bowtie \sigma_{stop \text{ IS NULL }}(O));$
- 4c.  $\pi_{S.indeks, nazwisko, rok}(S) \setminus \pi_{S.indeks, nazwisko, rok}(S \bowtie \sigma_{stop=5.0}(O));$

### Znaczenie zapytań

- 4a. Studenci, którzy dostali jakąś ocenę inną niż 5.0.
- 4b. Studenci, którzy nie dostali wpisu (niezgodne z więzami relacji)

## Przykład 4

### Baza danych

$S = (\underline{indeks}, nazwisko, rok)$ ,  $P = (\underline{nazwa}, typ)$ ,  $O = (\underline{indeks}, \underline{przed}, \underline{data}, stop)$

Studenci, którzy nie dostali 5.0.

- 4a.  $\pi_{S.indeks, nazwisko, rok}(S \bowtie \sigma_{stop \neq 5.0}(O));$
- 4b.  $\pi_{S.indeks, nazwisko, rok}(S \bowtie \sigma_{stop \text{ IS NULL }}(O));$
- 4c.  $\pi_{S.indeks, nazwisko, rok}(S) \setminus \pi_{S.indeks, nazwisko, rok}(S \bowtie \sigma_{stop=5.0}(O));$

### Znaczenie zapytań

- 4a. Studenci, którzy dostali jakąś ocenę inną niż 5.0.
- 4b. Studenci, którzy nie dostali wpisu (niezgodne z więzami relacji)
- 4c. Studenci, którzy nie dostali żadnej piątki.



## Przykład 4

### Baza danych

$S = (\underline{indeks}, nazwisko, rok)$ ,  $P = (\underline{nazwa}, typ)$ ,  $O = (\underline{indeks}, \underline{przed}, \underline{data}, stop)$

Studenci, którzy nie dostali 5.0.

- 4a.  $\pi_{S.indeks, nazwisko, rok}(S \bowtie \sigma_{stop \neq 5.0}(O));$
- 4b.  $\pi_{S.indeks, nazwisko, rok}(S \bowtie \sigma_{stop \text{ IS NULL } (O)});$
- 4c.  $\pi_{S.indeks, nazwisko, rok}(S) \setminus \pi_{S.indeks, nazwisko, rok}(S \bowtie \sigma_{stop = 5.0}(O));$
- 4d.  $\pi_{S.indeks, nazwisko, rok}(S) \setminus \pi_{S.indeks, nazwisko, rok}(S \bowtie \sigma_{stop \neq 5.0}(O));$

### Znaczenie zapytań

- 4a. Studenci, którzy dostali jakąś ocenę inną niż 5.0.
- 4b. Studenci, którzy nie dostali wpisu (niezgodne z więzami relacji)
- 4c. Studenci, którzy nie dostali żadnej piątki.

## Przykład 4

### Baza danych

$S = (\underline{indeks}, nazwisko, rok)$ ,  $P = (\underline{nazwa}, typ)$ ,  $O = (\underline{indeks}, \underline{przed}, \underline{data}, stop)$

Studenci, którzy nie dostali 5.0.

- 4a.  $\pi_{S.indeks, nazwisko, rok}(S \bowtie \sigma_{stop \neq 5.0}(O));$
- 4b.  $\pi_{S.indeks, nazwisko, rok}(S \bowtie \sigma_{stop \text{ IS NULL } (O)});$
- 4c.  $\pi_{S.indeks, nazwisko, rok}(S) \setminus \pi_{S.indeks, nazwisko, rok}(S \bowtie \sigma_{stop=5.0}(O));$
- 4d.  $\pi_{S.indeks, nazwisko, rok}(S) \setminus \pi_{S.indeks, nazwisko, rok}(S \bowtie \sigma_{stop \neq 5.0}(O));$

### Znaczenie zapytań

- 4a. Studenci, którzy dostali jakąś ocenę inną niż 5.0.
- 4b. Studenci, którzy nie dostali wpisu (niezgodne z więzami relacji)
- 4c. Studenci, którzy nie dostali żadnej piątki.
- 4d. Studenci, którzy mają tylko oceny 5.0 (być może nie mają żadnych).

## Przykład 4

### Baza danych

$S = (\underline{\text{indeks}}, \text{nazwisko}, \text{rok})$ ,  $P = (\underline{\text{nazwa}}, \text{typ})$ ,  $O = (\underline{\text{indeks}}, \underline{\text{przed}}, \underline{\text{data}}, \text{stop})$

Studenci, którzy nie dostali 5.0.

- 4a.  $\pi_{S.\text{indeks}, \text{nazwisko}, \text{rok}}(S \bowtie \sigma_{\text{stop} \neq 5.0}(O));$
- 4b.  $\pi_{S.\text{indeks}, \text{nazwisko}, \text{rok}}(S \bowtie \sigma_{\text{stop IS NULL}}(O));$
- 4c.  $\pi_{S.\text{indeks}, \text{nazwisko}, \text{rok}}(S) \setminus \pi_{S.\text{indeks}, \text{nazwisko}, \text{rok}}(S \bowtie \sigma_{\text{stop}=5.0}(O));$
- 4d.  $\pi_{S.\text{indeks}, \text{nazwisko}, \text{rok}}(S) \setminus \pi_{S.\text{indeks}, \text{nazwisko}, \text{rok}}(S \bowtie \sigma_{\text{stop} \neq 5.0}(O));$
- 4e.  $\pi_{S.\text{ind}, \text{naz}, \text{rok}}(S \bowtie O) \setminus \pi_{S.\text{indeks}, \text{nazwisko}, \text{rok}}(S \bowtie \sigma_{\text{stop} \neq 5.0}(O));$

### Znaczenie zapytań

- 4a. Studenci, którzy dostali jakąś ocenę inną niż 5.0.
- 4b. Studenci, którzy nie dostali wpisu (niezgodne z więzami relacji)
- 4c. Studenci, którzy nie dostali żadnej piątki.
- 4d. Studenci, którzy mają tylko oceny 5.0 (być może nie mają żadnych).

## Przykład 4

### Baza danych

$S = (\underline{\text{indeks}}, \text{nazwisko}, \text{rok})$ ,  $P = (\underline{\text{nazwa}}, \text{typ})$ ,  $O = (\underline{\text{indeks}}, \underline{\text{przed}}, \underline{\text{data}}, \text{stop})$

Studenci, którzy nie dostali 5.0.

- 4a.  $\pi_{S.\text{indeks}, \text{nazwisko}, \text{rok}}(S \bowtie \sigma_{\text{stop} \neq 5.0}(O));$
- 4b.  $\pi_{S.\text{indeks}, \text{nazwisko}, \text{rok}}(S \bowtie \sigma_{\text{stop IS NULL}}(O));$
- 4c.  $\pi_{S.\text{indeks}, \text{nazwisko}, \text{rok}}(S) \setminus \pi_{S.\text{indeks}, \text{nazwisko}, \text{rok}}(S \bowtie \sigma_{\text{stop}=5.0}(O));$
- 4d.  $\pi_{S.\text{indeks}, \text{nazwisko}, \text{rok}}(S) \setminus \pi_{S.\text{indeks}, \text{nazwisko}, \text{rok}}(S \bowtie \sigma_{\text{stop} \neq 5.0}(O));$
- 4e.  $\pi_{S.\text{ind}, \text{naz}, \text{rok}}(S \bowtie O) \setminus \pi_{S.\text{indeks}, \text{nazwisko}, \text{rok}}(S \bowtie \sigma_{\text{stop} \neq 5.0}(O));$

### Znaczenie zapytań

- 4a. Studenci, którzy dostali jakąś ocenę inną niż 5.0.
- 4b. Studenci, którzy nie dostali wpisu (niezgodne z więzami relacji)
- 4c. Studenci, którzy nie dostali żadnej piątki.
- 4d. Studenci, którzy mają tylko oceny 5.0 (być może nie mają żadnych).
- 4e. Studenci, którzy dostają tylko piątki, przy czym bierzemy pod uwagę tylko tych, którzy mają jakikolwiek wpis.

## Przykład 5

### Baza danych

$S = (\underline{indeks}, nazwisko, rok), P = (\underline{nazwa}, typ), O = (\underline{indeks}, \underline{przed}, \underline{data}, stop)$

Jak szukać czegoś, czego nie ma?

5a.  $\pi_{S.indeks, nazwisko}(S) \setminus \pi_{S.indeks, nazwisko}(S \bowtie O);$

## Przykład 5

### Baza danych

$S = (\underline{indeks}, nazwisko, rok), P = (\underline{nazwa}, typ), O = (\underline{indeks}, \underline{przed}, \underline{data}, stop)$

Jak szukać czegoś, czego nie ma?

5a.  $\pi_{S.indeks, nazwisko}(S) \setminus \pi_{S.indeks, nazwisko}(S \bowtie O);$

5b.  $\pi_{S.indeks, nazwisko}(S \bowtie \sigma_{stop \text{ IS NULL }}(O));$

## Przykład 5

### Baza danych

$S = (\underline{indeks}, nazwisko, rok)$ ,  $P = (\underline{nazwa}, typ)$ ,  $O = (\underline{indeks}, \underline{przed}, \underline{data}, stop)$

Jak szukać czegoś, czego nie ma?

5a.  $\pi_{S.indeks, nazwisko}(S) \setminus \pi_{S.indeks, nazwisko}(S \bowtie O)$ ;

5b.  $\pi_{S.indeks, nazwisko}(S \bowtie \sigma_{stop \text{ IS NULL }}(O))$ ;

5c.  $\pi_{S.indeks, nazwisko}(S \bowtie \sigma_{stop= \text{ NULL }}(O))$ ;

## Przykład 5

### Baza danych

$S = (\underline{indeks}, nazwisko, rok)$ ,  $P = (\underline{nazwa}, typ)$ ,  $O = (\underline{indeks}, \underline{przed}, \underline{data}, stop)$

Jak szukać czegoś, czego nie ma?

5a.  $\pi_{S.indeks, nazwisko}(S) \setminus \pi_{S.indeks, nazwisko}(S \bowtie O)$ ;

5b.  $\pi_{S.indeks, nazwisko}(S \bowtie \sigma_{stop \text{ IS NULL }}(O))$ ;

5c.  $\pi_{S.indeks, nazwisko}(S \bowtie \sigma_{stop= \text{ NULL }}(O))$ ;

5d.  $\pi_{S.indeks, nazwisko}(S \bowtie \sigma_{stop \neq \text{ NULL }}(O))$ ;



## Przykład 5

### Baza danych

$S = (\underline{indeks}, nazwisko, rok)$ ,  $P = (\underline{nazwa}, typ)$ ,  $O = (\underline{indeks}, \underline{przed}, \underline{data}, stop)$

Jak szukać czegoś, czego nie ma?

5a.  $\pi_{S.indeks, nazwisko}(S) \setminus \pi_{S.indeks, nazwisko}(S \bowtie O)$ ;

5b.  $\pi_{S.indeks, nazwisko}(S \bowtie \sigma_{stop \text{ IS NULL }}(O))$ ;

5c.  $\pi_{S.indeks, nazwisko}(S \bowtie \sigma_{stop= \text{ NULL }}(O))$ ;

5d.  $\pi_{S.indeks, nazwisko}(S \bowtie \sigma_{stop \neq \text{ NULL }}(O))$ ;

Krotka jest wybierana przez selekcję, gdy warunek ma dla niej wartość TRUE. Wartość UNKNOWN nie wystarcza.

# Przykład 6

## Baza danych

$S = (\underline{\text{indeks}}, \text{nazwisko}, \text{rok}), P = (\underline{\text{nazwa}}, \text{typ}), O = (\underline{\text{indeks}}, \underline{\text{przed}}, \underline{\text{data}}, \text{stop})$

Można pytać o to samo na różne sposoby. Czy to ma jakieś znaczenie?

$$(6) \pi_{\text{nazwisko}, \text{indeks}}(\sigma_{\text{stop}=5.0 \wedge \text{typ}='zaaw'}(\sigma_{\text{nazwa}=\text{przed}}(P \times O)) \bowtie \sigma_{\text{rok}=4}(S))$$

$$\cup \pi_{\text{nazwisko}, \text{indeks}}(\sigma_{\text{stop}=5.0 \wedge \text{typ}='obow'}(\sigma_{\text{nazwa}=\text{przed}}(P \times O)) \bowtie \sigma_{\text{rok}=3}(S));$$

$$(6a) \pi_{\text{nazwisko}, \text{indeks}}(\sigma_{((\text{rok}=3 \wedge \text{typ}='obow') \vee (\text{rok}=4 \wedge \text{typ}='zaaw'))}(\sigma_{\text{rok}=3 \vee \text{rok}=4}(S) \bowtie \pi_{\text{indeks}, \text{typ}}(\rho_{P(\text{przed}, \text{typ})}(\sigma_{\text{typ}='zaaw' \vee \text{typ}='obow'}(P))) \bowtie \pi_{\text{indeks}, \text{przed}}(\sigma_{\text{stop}=5.0}(O))))$$

## Przykład 7 - poszukajmy najlepszych z BD

### Baza danych

$S = (\underline{\text{indeks}}, \text{nazwisko}, \text{rok}), P = (\underline{\text{nazwa}}, \text{typ}), O = (\underline{\text{indeks}}, \underline{\text{przed}}, \underline{\text{data}}, \text{stop})$

## Przykład 7 - poszukajmy najlepszych z BD

### Baza danych

$S = (\underline{\text{indeks}}, \text{nazwisko}, \text{rok}), P = (\underline{\text{nazwa}}, \text{typ}), O = (\underline{\text{indeks}}, \underline{\text{przed}}, \underline{\text{data}}, \text{stop})$

(7a)  $\pi_{\text{indeks}}(\sigma_{\text{stop} > s1 \wedge \text{przed} = \text{'BD'}} \wedge p1 = \text{przed}(O \bowtie \rho_{O1(i1, p1, d1, s1)}(O)))$

### Znaczenie zapytań

## Przykład 7 - poszukajmy najlepszych z BD

### Baza danych

$S = (\underline{indeks}, nazwisko, rok), P = (\underline{nazwa}, typ), O = (\underline{indeks}, \underline{przed}, \underline{data}, stop)$

$$(7a) \pi_{indeks}(\sigma_{stop > s1 \wedge przed = 'BD' \wedge p1 = przed}(O \bowtie \rho_{O1(i1, p1, d1, s1)}(O)))$$

### Znaczenie zapytań

- 7a. Indeksy studentów, którzy z BD mają ocenę lepszą niż ktoś inny, czyli nie są najgorsi.

## Przykład 7 - poszukajmy najlepszych z BD

### Baza danych

$S = (\underline{\text{indeks}}, \text{nazwisko}, \text{rok}), P = (\underline{\text{nazwa}}, \text{typ}), O = (\underline{\text{indeks}}, \underline{\text{przed}}, \underline{\text{data}}, \text{stop})$

$$(7a) \pi_{\text{indeks}}(\sigma_{\text{stop} > s1 \wedge \text{przed} = \text{'BD'}} \wedge p1 = \text{przed}(O \bowtie \rho_{O1}(i1, p1, d1, s1)(O)))$$

### Znaczenie zapytań

- 7a. Indeksy studentów, którzy z BD mają ocenę lepszą niż ktoś inny, czyli nie są najgorsi.
- 7b. Indeksy studentów, którzy z BD mają ocenę gorszą niż ktoś inny, czyli nie są najlepsi (dopełnienie tego, czego szukamy).

## Przykład 7 - poszukajmy najlepszych z BD

### Baza danych

$S = (\underline{\text{indeks}}, \text{nazwisko}, \text{rok}), P = (\underline{\text{nazwa}}, \text{typ}), O = (\underline{\text{indeks}}, \underline{\text{przed}}, \underline{\text{data}}, \text{stop})$

(7a)  $\pi_{\text{indeks}}(\sigma_{\text{stop} > s1 \wedge \text{przed} = \text{'BD'}} \wedge p1 = \text{przed}(O \bowtie \rho_{O1}(i1, p1, d1, s1)(O)))$

(7b)  $\pi_{\text{indeks}}(\sigma_{\text{stop} < s1 \wedge \text{przed} = \text{'BD'}} \wedge p1 = \text{przed}(O \bowtie \rho_{O1}(i1, p1, d1, s1)(O)))$

### Znaczenie zapytań

- 7a. Indeksy studentów, którzy z BD mają ocenę lepszą niż ktoś inny, czyli nie są najgorsi.
- 7b. Indeksy studentów, którzy z BD mają ocenę gorszą niż ktoś inny, czyli nie są najlepsi (dopełnienie tego, czego szukamy).

## Przykład 7 - poszukajmy najlepszych z BD

### Baza danych

$S = (\underline{\text{indeks}}, \text{nazwisko}, \text{rok}), P = (\underline{\text{nazwa}}, \text{typ}), O = (\underline{\text{indeks}}, \underline{\text{przed}}, \underline{\text{data}}, \text{stop})$

$$(7a) \pi_{\text{indeks}}(\sigma_{\text{stop} > s1 \wedge \text{przed} = \text{"BD"}} \wedge p1 = \text{przed}(O \bowtie \rho_{O1}(i1, p1, d1, s1)(O)))$$

$$(7b) \pi_{\text{indeks}}(\sigma_{\text{stop} < s1 \wedge \text{przed} = \text{"BD"}} \wedge p1 = \text{przed}(O \bowtie \rho_{O1}(i1, p1, d1, s1)(O)))$$

$$(7c) \pi_{\text{indeks}}(S) \setminus \pi_{\text{indeks}}(\sigma_{\text{stop} < s1 \wedge \text{przed} = \text{"BD"}} \wedge p1 = \text{przed}(O \bowtie \rho_{O1}(i1, p1, d1, s1)(O)))$$

### Znaczenie zapytań

- 7a. Indeksy studentów, którzy z BD mają ocenę lepszą niż ktoś inny, czyli nie są najgorsi.
- 7b. Indeksy studentów, którzy z BD mają ocenę gorszą niż ktoś inny, czyli nie są najlepsi (dopełnienie tego, czego szukamy).



## Przykład 7 - poszukajmy najlepszych z BD

### Baza danych

$S = (\underline{\text{indeks}}, \text{nazwisko}, \text{rok}), P = (\underline{\text{nazwa}}, \text{typ}), O = (\underline{\text{indeks}}, \underline{\text{przed}}, \underline{\text{data}}, \text{stop})$

$$(7a) \pi_{\text{indeks}}(\sigma_{\text{stop} > s1 \wedge \text{przed} = \text{"BD"}} \wedge p1 = \text{przed}(O \bowtie \rho_{O1(i1, p1, d1, s1)}(O)))$$

$$(7b) \pi_{\text{indeks}}(\sigma_{\text{stop} < s1 \wedge \text{przed} = \text{"BD"}} \wedge p1 = \text{przed}(O \bowtie \rho_{O1(i1, p1, d1, s1)}(O)))$$

$$(7c) \pi_{\text{indeks}}(S) \setminus \pi_{\text{indeks}}(\sigma_{\text{stop} < s1 \wedge \text{przed} = \text{"BD"}} \wedge p1 = \text{przed}(O \bowtie \rho_{O1(i1, p1, d1, s1)}(O)))$$

### Znaczenie zapytań

- 7a. Indeksy studentów, którzy z BD mają ocenę lepszą niż ktoś inny, czyli nie są najgorsi.
- 7b. Indeksy studentów, którzy z BD mają ocenę gorszą niż ktoś inny, czyli nie są najlepsi (dopełnienie tego, czego szukamy).
- 7c. Indeksy studentów, którzy nie są od nikogo gorsi z BD.

## Przykład 7 - poszukajmy najlepszych z BD

### Baza danych

$S = (\underline{\text{indeks}}, \text{nazwisko}, \text{rok}), P = (\underline{\text{nazwa}}, \text{typ}), O = (\underline{\text{indeks}}, \underline{\text{przed}}, \underline{\text{data}}, \text{stop})$

$$(7a) \pi_{\text{indeks}}(\sigma_{\text{stop} > s1 \wedge \text{przed} = \text{"BD"}} \wedge p1 = \text{przed}(O \bowtie \rho_{O1}(i1, p1, d1, s1)(O)))$$

$$(7b) \pi_{\text{indeks}}(\sigma_{\text{stop} < s1 \wedge \text{przed} = \text{"BD"}} \wedge p1 = \text{przed}(O \bowtie \rho_{O1}(i1, p1, d1, s1)(O)))$$

$$(7c) \pi_{\text{indeks}}(S) \setminus \pi_{\text{indeks}}(\sigma_{\text{stop} < s1 \wedge \text{przed} = \text{"BD"}} \wedge p1 = \text{przed}(O \bowtie \rho_{O1}(i1, p1, d1, s1)(O)))$$

$$(7d) \pi_{\text{indeks}}(\sigma_{\text{przed} = \text{"BD"}}(O)) \setminus \pi_{\text{indeks}}(\sigma_{\text{stop} < s1 \wedge \text{przed} = \text{"BD"}} \wedge p1 = \text{przed}(O \bowtie \rho_{O1}(i1, p1, d1, s1)(O)))$$

### Znaczenie zapytań

- 7a. Indeksy studentów, którzy z BD mają ocenę lepszą niż ktoś inny, czyli nie są najgorsi.
- 7b. Indeksy studentów, którzy z BD mają ocenę gorszą niż ktoś inny, czyli nie są najlepsi (dopełnienie tego, czego szukamy).
- 7c. Indeksy studentów, którzy nie są od nikogo gorsi z BD.

## Przykład 7 - poszukajmy najlepszych z BD

### Baza danych

$S = (\underline{\text{indeks}}, \text{nazwisko}, \text{rok}), P = (\underline{\text{nazwa}}, \text{typ}), O = (\underline{\text{indeks}}, \underline{\text{przed}}, \underline{\text{data}}, \text{stop})$

$$(7a) \pi_{\text{indeks}}(\sigma_{\text{stop} > s1 \wedge \text{przed} = \text{"BD"}} \wedge p1 = \text{przed}(O \bowtie \rho_{O1}(i1, p1, d1, s1)(O)))$$

$$(7b) \pi_{\text{indeks}}(\sigma_{\text{stop} < s1 \wedge \text{przed} = \text{"BD"}} \wedge p1 = \text{przed}(O \bowtie \rho_{O1}(i1, p1, d1, s1)(O)))$$

$$(7c) \pi_{\text{indeks}}(S) \setminus \pi_{\text{indeks}}(\sigma_{\text{stop} < s1 \wedge \text{przed} = \text{"BD"}} \wedge p1 = \text{przed}(O \bowtie \rho_{O1}(i1, p1, d1, s1)(O)))$$

$$(7d) \pi_{\text{indeks}}(\sigma_{\text{przed} = \text{"BD"}}(O)) \setminus \pi_{\text{indeks}}(\sigma_{\text{stop} < s1 \wedge \text{przed} = \text{"BD"}} \wedge p1 = \text{przed}(O \bowtie \rho_{O1}(i1, p1, d1, s1)(O)))$$

### Znaczenie zapytań

- 7a. Indeksy studentów, którzy z BD mają ocenę lepszą niż ktoś inny, czyli nie są najgorsi.
- 7b. Indeksy studentów, którzy z BD mają ocenę gorszą niż ktoś inny, czyli nie są najlepsi (dopełnienie tego, czego szukamy).
- 7c. Indeksy studentów, którzy nie są od nikogo gorsi z BD.
- 7d. Indeksy studentów, którzy są najlepsi z BD.

## Przykład 7 - poszukajmy najlepszych z BD

### Baza danych

$S = (\underline{\text{indeks}}, \text{nazwisko}, \text{rok}), P = (\underline{\text{nazwa}}, \text{typ}), O = (\underline{\text{indeks}}, \underline{\text{przed}}, \underline{\text{data}}, \text{stop})$

$$(7a) \pi_{\text{indeks}}(\sigma_{\text{stop} > s1 \wedge \text{przed} = \text{"BD"}} \wedge p1 = \text{przed}(O \bowtie \rho_{O1}(i1, p1, d1, s1)(O)))$$

$$(7b) \pi_{\text{indeks}}(\sigma_{\text{stop} < s1 \wedge \text{przed} = \text{"BD"}} \wedge p1 = \text{przed}(O \bowtie \rho_{O1}(i1, p1, d1, s1)(O)))$$

$$(7c) \pi_{\text{indeks}}(S) \setminus \pi_{\text{indeks}}(\sigma_{\text{stop} < s1 \wedge \text{przed} = \text{"BD"}} \wedge p1 = \text{przed}(O \bowtie \rho_{O1}(i1, p1, d1, s1)(O)))$$

$$(7d) \pi_{\text{indeks}}(\sigma_{\text{przed} = \text{"BD"}}(O)) \setminus \pi_{\text{indeks}}(\sigma_{\text{stop} < s1 \wedge \text{przed} = \text{"BD"}} \wedge p1 = \text{przed}(O \bowtie \rho_{O1}(i1, p1, d1, s1)(O)))$$

$$? \sigma_{\text{indeks} \neq i1}(O \bowtie \rho_{O1}(i1, p1, d1, s1)(O))$$

### Znaczenie zapytań

- 7a. Indeksy studentów, którzy z BD mają ocenę lepszą niż ktoś inny, czyli nie są najgorsi.
- 7b. Indeksy studentów, którzy z BD mają ocenę gorszą niż ktoś inny, czyli nie są najlepsi (dopełnienie tego, czego szukamy).
- 7c. Indeksy studentów, którzy nie są od nikogo gorsi z BD.
- 7d. Indeksy studentów, którzy są najlepsi z BD.

## Wnioski i uwagi:

- 1 Algebra relacji jest językiem imperatywnym (operacyjnym).

## Wnioski i uwagi:

- 1 Algebra relacji jest językiem imperatywnym (operacyjnym).
- 2 Znaczenie zapytania (w języku naturalnym) nie zawsze jest oczywiste, gdyż algebra relacji nie przypomina języka naturalnego.

## Wnioski i uwagi:

- 1 Algebra relacji jest językiem imperatywnym (operacyjnym).
- 2 Znaczenie zapytania (w języku naturalnym) nie zawsze jest oczywiste, gdyż algebra relacji nie przypomina języka naturalnego.
- 3 To samo zapytanie może mieć wiele równoważnych postaci — mogą one różnić się złożonością wykonania.

## Wnioski i uwagi:

- 1 Algebra relacji jest językiem imperatywnym (operacyjnym).
- 2 Znaczenie zapytania (w języku naturalnym) nie zawsze jest oczywiste, gdyż algebra relacji nie przypomina języka naturalnego.
- 3 To samo zapytanie może mieć wiele równoważnych postaci — mogą one różnić się złożonością wykonania.
- 4 Na podstawie samego opisu trudno określić moc tego języka.



## Wnioski i uwagi:

- 1 Algebra relacji jest językiem imperatywnym (operacyjnym).
- 2 Znaczenie zapytania (w języku naturalnym) nie zawsze jest oczywiste, gdyż algebra relacji nie przypomina języka naturalnego.
- 3 To samo zapytanie może mieć wiele równoważnych postaci — mogą one różnić się złożonością wykonania.
- 4 Na podstawie samego opisu trudno określić moc tego języka.
- 5 Algebra relacji jest podstawą SQL.

## Materiały na skosie:

- 1 [https://skos.ii.uni.wroc.pl/pluginfile.php/20517/mod\\_label/intro/algebra\\_relacji\\_trudne\\_przyklady\\_EU.pdf](https://skos.ii.uni.wroc.pl/pluginfile.php/20517/mod_label/intro/algebra_relacji_trudne_przyklady_EU.pdf)

## Materiały na skosie:

- 1 [https://skos.ii.uni.wroc.pl/pluginfile.php/20517/mod\\_label/intro/algebra\\_relacji\\_trudne\\_przyklady\\_EU.pdf](https://skos.ii.uni.wroc.pl/pluginfile.php/20517/mod_label/intro/algebra_relacji_trudne_przyklady_EU.pdf)
- 2 <https://dbis-uibk.github.io/relax/>