Integralność danych Wersje języka SQL Klauzula SELECT i JOIN

Robert A. Kłopotek

r.klopotek@uksw.edu.pl

Wydział Matematyczno-Przyrodniczy. Szkoła Nauk Ścisłych, UKSW

Integralność danych

- Aspekty integralności
  - Integralność encji
  - Integralność powiązań
  - Dodatkowe ograniczenia
- Współczesne SZBD (DBMS), jeśli są poprawnie skonfigurowane, nie dopuszczają instrukcji modyfikacji danych, która narusza integralność danych

Integralność encji

- CEL: zagwarantowanie unikalności klucza głównego przez wszystkie rekordy w tabeli
- Rozwiązania:
  - Unikalny klucz główny (PK) jest zdefiniowany dla tabeli (tworzony jest również unikalny indeks)
  - SZBD (DBMS) monitoruje operacje CRUD (create/update/delete) aby zachować unikalność klucza głównego (PK)

Integralność powiązań

- CEL: zagwarantowanie, że klucze obce wskazują na istniejące rekordy identyfikowane przez odpowiadające im klucze główne
- Rozwiązania:
  - Klucz główny jest zdefiniowany dla tabeli zawierającej klucz główny + klucz obcy jest zdefiniowany dla tabeli zawierającej klucz obcy aby ustanowić relację
  - SZBD (DBMS) monitoruje operacje CRUD (create/update/delete) aby zachować/podtrzymywać relacje

Integralność powiązań - przykład

Sale

Budynek	Piętro	Sala
WOY/21	0	033
WOY/21	1	134
…		…

Wyposażenie

Budynek	Zasób	Sala
WOY/21	Krzesło	033
WOY/21	Stolik	033

klucz główny (PK – primary key)

klucz obcy (FK – foreign key)

- Każde wyposażenie musi wskazywać na istniejącą salę w tabeli Sale
- SZBD (DBMS) nie pozwoli na utworzenie wyposażenia, które znajduje się w nieistniejącej sali

Dodatkowe ograniczenia

- Istnieje możliwość zdefiniowania warunków logicznych, które musza być spełnione przez wszystkie rekordy, np.:
  - Pensja > podatek aby zapewnić, że wartość w kolumnie pensja jest większa od wartości podatku dla każdej osoby w tabeli
- Te warunki są sprawdzane przy każdej operacji CRUD (create/update/delete) wykonywanej na tabeli

Język SQL

- SQL – Structured Query Language – to język używany do:
  - Definiowania struktur danych
  - Aktualizowania informacji
  - Zapytań o informację
- Istnieją różne wersje standardu SQL:
  - SQL86 a.k.a. SQL1
  - SQL92 a.k.a. SQL2
  - SQL:1999 a.k.a SQL3 – wyrażenia regularne, kwerendy rekurencyjne, trigery, pewne cechy obiektowości (struktury)
  - SQL:2003 – podstawowa obsługa XML, kolumny z automatycznie generowanymi wartościami
  - SQL:2006 – pełna obsługa XML
  - SQL:2008 – instrukcja „TRUNCATE TABLE” ekwiwalent „DELETE FROM mytable”
  - SQL:2013 – tymczasowe klucze główne, tymczasowa integralność powiązań

Implementacje SQL

- Na początku standard SQL bazował na wspólnym zbiorze funkcjonalności zaimplementowanych w systemach bazodanowych IBM i Oracle
- Standard tylko częściowo zaimplementowany w SZBD innych producentów
- Dlatego napisanie kodu, który jest przenośny pomiędzy różnymi SZBD (DBMS) jest sztuką!
- Różni producenci SZBD oferują inne rozszerzenia języka (dialekty języka) zawierające instrukcje procedur jak PL/SQL firmy Oracle czy Transact-SQL Microsoftu

Instrukcja SQL SELECT

- Instrukcja SELECT służy do wyszukiwania informacji w tabeli lub zbiorze tabel połączonych relacją (klucze główne i klucze obce)

SQL SELECT

Typ_sali	Budynek	Piętro	Sala
laboratorium	WOY/21	0	033
wykładowa	WOY/21	3	314
	…		…
laboratorium	WOY/12	1	1221

```
SELECT * FROM Sale

SELECT budynek, sala FROM Sale
```

SELECT aliasy

```
SELECT budynek AS bud, sala AS sa
FROM Sale
```

Wynik zapytania:

bud	sa
WOY/21	033
WOY/21	314
…	…
WOY/12	1221

SELECT – instrukcja WHERE

```
SELECT * FROM Sale
WHERE pietro >2 AND typ_sali = ’laboratorium’
```

Wynik zapytania:

Typ_sali	Budynek	Piętro	Sala
wykładowa	WOY/21	3	314
	…		…

SELECT – fraza IN

```
SELECT * FROM Sale WHERE pietro IN (2,3)
```

Wynik zapytania:

Typ_sali	Budynek	Piętro	Sala
laboratorium	WOY/12	2	1242
	…		…
wykładowa	WOY/21	3	314

SELECT – instrukcja ORDER BY

```
SELECT * FROM Sale
WHERE pietro >=1 ORDER BY typ_sali, piętro
```

Wynik zapytania:

Typ_sali	Budynek	Piętro	Sala
laboratorium	WOY/12	1	1221
wykładowa	WOY/21	2	214
	…		…
wykładowa	WOY/21	3	314

SELECT – instrukcja ORDER BY

```
SELECT * FROM rooms [. . .] ORDER BY
Column_Name [DESC] [,Column_Name. . .]
```

- Jest możliwe , aby sortować w porządku malejącym (DESC)
- Domyślny jest porządek rosnący
- Uwaga praktyczna: kolumny zawijające tekst są sortowane jak tekst, np. jeśli kolumna zawiera numery jako test to ’1000’ < ’ 2’
- Kiedy używamy do sortowania wielu kolumn to najpierw jest sortowanie po pierwszej kolumnie, potem po drugiej itd.

SELECT – fraza DISTINCT

```
SELECT DISTINCT typ_sali, piętro FROM Sale
WHERE pietro >=2
```

Wynik zapytania:

Typ_sali	Piętro
laboratorium	1
wykładowa	2
wykładowa	3

Otrzymamy tylko unikalne kombinacje kolumn

Łączenia tabel oraz iloczyn kartezjański

Sale

Typ_sali	Budynek	Piętro	Sala
laboratorium	WOY/12	1	1221
wykładowa	WOY/21	2	214

Wyposażenie

Budynek	Zasób	Sala
WOY/12	Krzesło	1221
WOY/12	Stolik	1221

W jaki sposób wykonać zapytanie obejmujące wiele tablic, np. liczba zasobów w budynku WOY/21 na piętrze 1?

Łączenia tabel – style zapytań

OLD-STYLE QUERY

```
SELECT … FROM
TABLE1,TABLE2[,…]
WHERE Condition1[ AND
Condition2 …] […]
```

Przykład:

```
SELECT * FROM Sale,
Wyposażenie WHERE
Sale.Budynek =
Wyposażenie.Budynek AND
Sale.sala=
Wyposażenie.sala
```

MODERN-STYLE QUERY

```
SELECT … FROM TABLE1
JOIN TABLE2[ JOIN …]
ON Condition1[ AND
Condition2 …] […]
```

Przykład:

```
SELECT * FROM Sale
JOIN Wyposażenie ON
Sale.Budynek =
Wyposażenie.Budynek AND
Sale.sala =
Wyposażenie.sala
```

Interpretacja zapytania – krok 1

Otrzymujemy iloczyn kartezjański

Budynek_A	Piętro	Sala	Budynek_B	Zasób	Sala
WOY/12	1	1221	WOY/12	Krzesło	1221
WOY/21	2	214	WOY/12	Krzesło	1221
WOY/12	1	1221	WOY/12	Stolik	1221
WOY/21	2	214	WOY/12	Stolik	1221

Aby odpowiedzieć na zapytanie obejmujące wiele tabel tworzony jest najpierw iloczyn kartezjański np. tworzona jest tymczasowa tablica wszystkich możliwych kombinacji rekordów.

Interpretacja zapytania – krok 2

Otrzymujemy iloczyn kartezjański

Budynek_A	Piętro	Sala	Budynek_B	Zasób	Sala
WOY/12	1	1221	WOY/12	Krzesło	1221
WOY/12	1	1221	WOY/12	Stolik	1221

W drugim kroku iloczyn kartezjański jest filtrowany tak, aby zawierał tylko wiersze, które spełniają warunek:

```
Sale.Budynek = Wyposażenie.Budynek
AND Sale.sala= Wyposażenie.sala
```

Zapytania do wielu tabel (Cross-table queries)

- Unikalność kolumn w otrzymanym zbiorze jest zapewniana przez SZBD (DBMS). Stąd, jeśli jest to konieczne, dodawany jest sufiks np. Budynek_A, Budynek_B
- Jeśli musimy odwoływać się do jakiejś kolumny z zapytania to:
  - Jeśli nazwa kolumny jest unikalna w zapytaniu to wystarczy użycie jej nazwy (unqualified name), np. `piętro`
  - W przeciwnym razie należy użyć pełnej nazwy (qualified name), np. `Sale.Budynek`

Inne rodzaje JOIN

Sale

Typ_sali	Budynek	Piętro	Sala
laboratorium	WOY/12	1	1221
wykładowa	WOY/21	2	214

Wyposażenie

Budynek	Zasób	Sala
WOY/12	Krzesło	1221
WOY/12	Stolik	1221

W jaki sposób otrzymać sale 214 w wyniku jeśli nie ma żadnego wyposażenia w tej sali?

Inne rodzaje JOIN

Sale

Typ_sali	Budynek	Piętro	Sala
laboratorium	WOY/12	1	1221
wykładowa	WOY/21	2	214

Wyposażenie

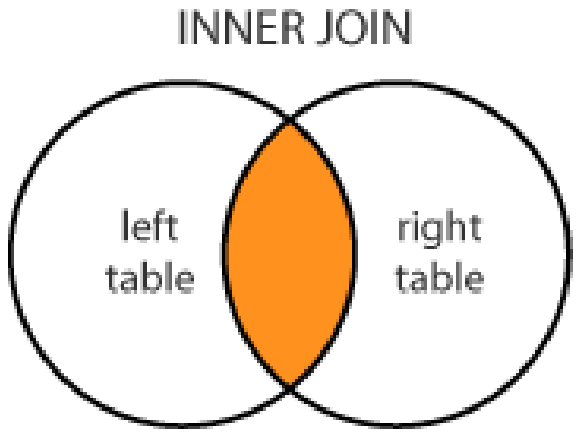
Budynek	Zasób	Sala
WOY/12	Krzesło	1221
WOY/12	Stolik	1221

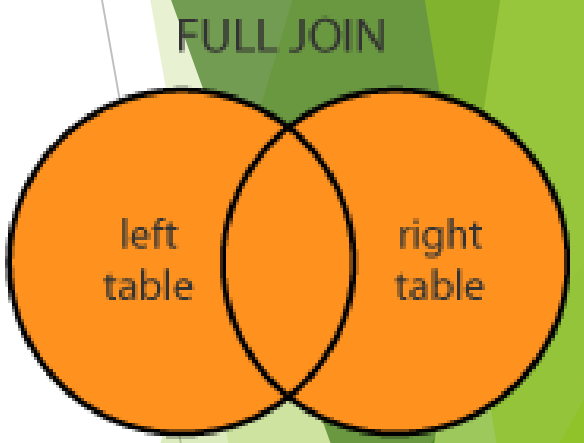
MODERN-STYLE QUERY

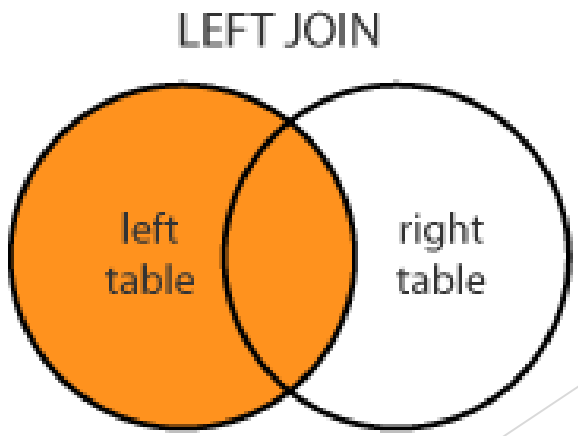
```
SELECT … FROM TABLE1
[INNER|LEFT|RIGHT|FULL]
JOIN TABLE2[ JOIN …]
ON Condition1[ AND
Condition2 …] […]
```

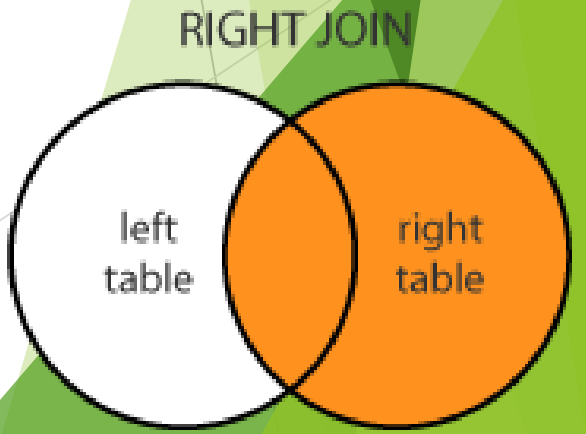
Inne rodzaje JOIN

- **INNER JOIN**: Zwraca takie rekordy, że istnieje co najmniej jedno dopasowanie w OBU tabelach
- **LEFT JOIN**: Zwraca wszystkie rekordy z lewej tabeli oraz dopasowane rekordy z prawej tabeli
- **RIGHT JOIN**: Zwraca wszystkie rekordy z prawej tabeli oraz dopasowane rekordy z lewej tabeli
- **FULL JOIN**: Zwraca takie rekordy, że istnieje co najmniej jedno dopasowanie w JEDNEJ z tabel







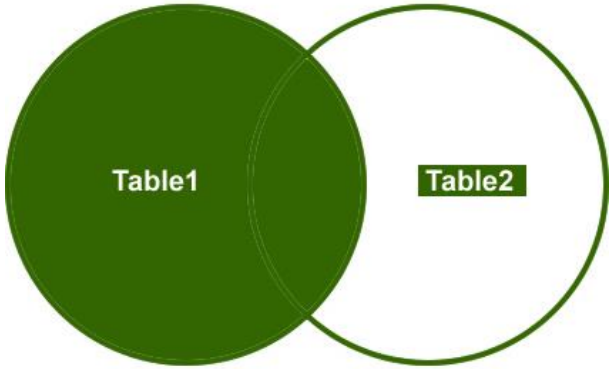


JOIN w MySQL

- JOIN, CROSS JOIN, i INNER JOIN są ekwiwalentami (mogą być sobą zastępowane). W standardowym SQL CROSS JOIN jest iloczynem kartezjańskim
- NATURAL [LEFT] JOIN jest ekwiwalentem odpowiednio INNER JOIN i LEFT JOIN
- RIGHT JOIN konwertowany do LEFT JOIN z uwzględnieniem zamiany tabel. Ze względu na kompatybilność zalecane jest używanie LEFT JOIN
- STRAIGHT_JOIN działa jak JOIN, ale lewa tabela jest czytana jako pierwsza. Może być to przydatne, jeśli optymalizator bazy danych źle układa tabele
- OUTER JOIN (lub FULL OUTER JOIN) – zwraca wszystkie rekordy niezależnie od dopasowania. W miejsca niedopasowane wstawiana jest wartość NULL

JOIN w MS SQL (T-SQL)

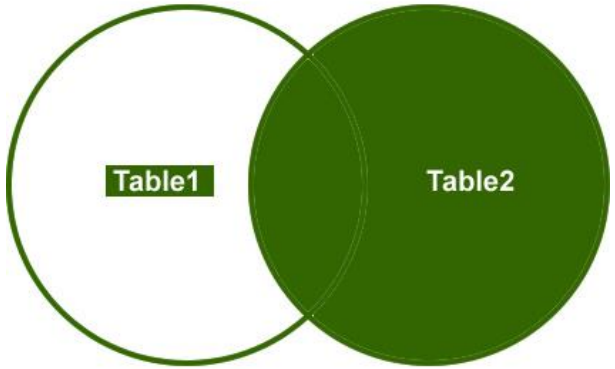
LEFT OUTER JOIN




```
SELECT *
FROM Table1 t1
LEFT OUTER JOIN Table2 t2
        ON t1.Col1 = t2.Col1
```

(C) http://blog.SQLAuthority.com

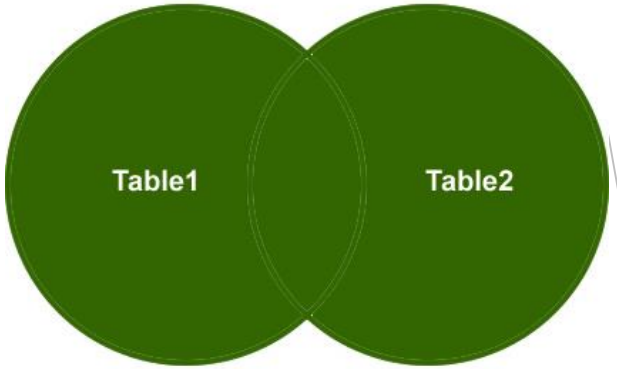
RIGHT OUTER JOIN




```
SELECT *
FROM Table1 t1
RIGHT OUTER JOIN Table2 t2
        ON t1.Col1 = t2.Col1
```

(C) http://blog.SQLAuthority.com

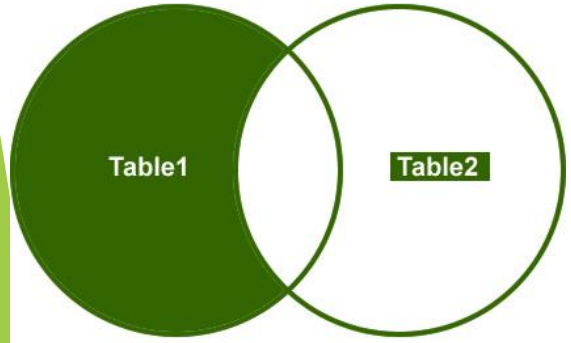
FULL OUTER JOIN




```
SELECT *
FROM Table1 t1
FULL OUTER JOIN Table2 t2
        ON t1.Col1 = t2.Col1
```

(C) http://blog.SQLAuthority.com

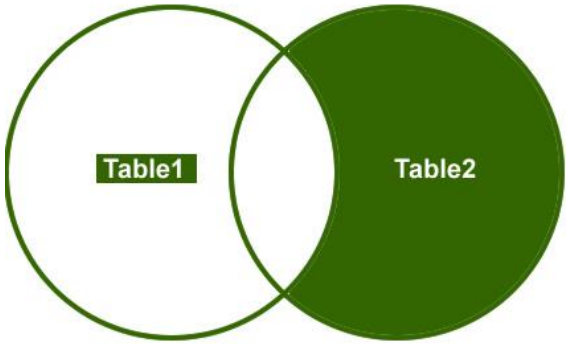
LEFT OUTER JOIN - WHERE NULL




```
SELECT *
FROM Table1 t1
LEFT OUTER JOIN Table2 t2
        ON t1.Col1 = t2.Col1
WHERE t2.Col1 IS NULL
```

(C) http://blog.SQLAuthority.com

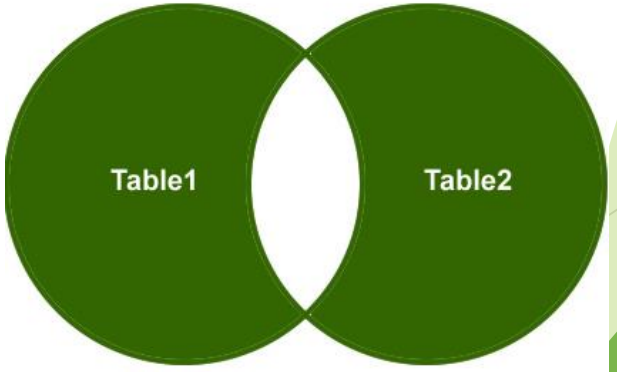
RIGHT OUTER JOIN - WHERE NULL




```
SELECT *
FROM Table1 t1
RIGHT OUTER JOIN Table2 t2
        ON t1.Col1 = t2.Col1
WHERE t1.Col1 IS NULL
```

(C) http://blog.SQLAuthority.com

OUTER JOIN - WHERE NULL




```
SELECT *
FROM Table1 t1
FULL OUTER JOIN Table2 t2
        ON t1.Col1 = t2.Col1
WHERE t1.Col1 IS NULL
        OR t2.Col1 IS NULL
```

(C) http://blog.SQLAuthority.com

T-SQL CROSS JOIN

	ID	Value
1	1	First
2	2	Second
3	3	Third
4	4	Fourth
5	5	Fifth

	ID	Value
1	1	First
2	2	Second
3	3	Third
4	6	Sixth
5	7	Seventh
6	8	Eighth

	ID	Value	ID	Value
1	1	First	1	First
2	1	First	2	Second
3	1	First	3	Third
4	1	First	6	Sixth
5	1	First	7	Seventh
6	1	First	8	Eighth
7	2	Second	1	First
8	2	Second	2	Second
9	2	Second	3	Third
10	2	Second	6	Sixth
11	2	Second	7	Seventh
12	2	Second	8	Eighth

5*6=30 rekordów

Pytania?