Bazy danych

mgr inż. Nikodem Bulanda

19 kwietnia 2023

Streszczenie

Instalacja, konfiguracja i obsługa systemu zarządzania bazą danych MySQL 8.0. Ćwiczenia podstawowe, realizowaną w oparciu o oficjalną dokumentację.

Spis treści

1 Uruchomienie serwera SZDB MySQL 8.0

Niniejszy rozdział stanowić będzie skompensowany przewodnik po instalacji i uruchomieniu SZDB MySQL 8.0. Przybliżone zostaną w nim sposoby uruchomienia, zarządzania i konserwacji systemu bazodanowego. Pełny opis działań i dokumentacje znajdziecie w poniższym linku.

Dokumentacja ! Przeczytaj i zapoznaj się z dokumentacją
`https://dev.mysql.com/doc/refman/8.0/en/windows-install-archive.html`

1.1 Podstawowa konfiguracja

1. Pobierz archiwum ZIP SZBD mysql w wersji 8.0 - *"no debug'*
   `https://dev.mysql.com/downloads/mysql/`
2. Zawartość archiwum wypakuj do katalogu *C:\ISNS\mysql*.
   [UWAGA!] Pamiętaj aby przenieść tam zawartość archiwum a nie folder powstały po jego wypakowaniu
3. Stwórz plik konfiguracyjny i zapisz go w katalogu *C:\ISNS\mysql*

```
[mysqld]
# set basedir to your installation path
basedir=C:\ISNS\mysql
# set datadir to the location of your data directory
datadir=C:\ISNS\mysql\data
```

Listing 1: Plik konfiguracyjny my.ini - lokalizacja basedir

1.2 Inicjalizacja

Po zainstalowaniu[1] MySQL, konieczne będzie przeprowadzanie procesu inicjalizacji. W skład czynności inicjalizacyjnych wchodzić będzie między innymi:

- utworzenie katalogu *'data'*,
- utworzenie bazy konfiguracyjnej "mysql"[2],
- utworzenie domyślnych kont użytkowników,
- deklaracja schematu domyślnego itp.

Inicjalizacja może odbywać się w dwóch trybach, zabezpieczonym i niezabezpieczonym. Tryby te odróżnia poziom domyślnie ustalanych zabezpieczeń. W trybie *–initialize-insecure*, użytkownikowi root zostaje nadane domyślnie **puste** hasło.

[1] w naszym przypadku osadzeniu plików

[2] ze względu na to iż SZDB MySQL jest systemem przechowywania informacji, nie dziwić powinien fakt iż większość ustawień przechowywanych jest właśnie w bazie

[UWAGA!] Proces inicjalizacji wykonaj tylko raz !!!
[UWAGA!] Polecenie wykonaj z poziomu katalogu mysql "basedir".

```
bin\mysqld --initialize-insecure --user=mysql

lub

bin\mysqld --initialize-insecure --user=mysql
--defaults-file = C:\ISNS\mysql\my.ini
```

Listing 2: Inicjalizacja SZDB MySQL 8.X

1.3 Uruchomienie serwera

Uruchomienie najlepiej przeprowadzać z wiersza poleceń w oknie konsoli (lub „ oknie DOS/CMD ”). W ten sposób możesz wyświetlać komunikaty o stanie serwera w oknie wiersza poleceń, w którym był on uruchamiany. Komunikaty te ułatwiają identyfikację i naprawę problemów w razie ich wystąpienia. Aby uruchomić serwer, wpisz to polecenie:

```
bin\mysqld --defaults-file = C:\ISNS\mysql\my.ini --console

lub

bin\mysqld
```

Listing 3: Uruchomienie serwera bazy danych

W przypadku pominięcia opcji *–console* serwer zapisuje[3] dane diagnostyczne w dzienniku błędów wskazanym w konfiguracji lub w deklaracji '*in-line*' podczas uruchamiania *mysqld*.

1.4 Zatrzymanie serwera

Narzędzie administracyjne MySQL **mysqladmin.exe**, pozwala połączyć się z serwerem i nakazać mu zamknięcie. Narzędzie to łączy się z serwerem jako użytkownik ***root*** MySQL, (który jest domyślnym kontem administracyjnym w systemie MySQL). Jeśli konto użytkownika ***root*** MySQL posiada hasło, musisz wywołać mysqladmin z dodatkowym przełącznikiem '-p' i podać hasło, gdy zostaniesz o to poproszony.

```
bin\mysqladmin -u root shutdown

lub

bin\mysqladmin -u root -p shutdown
```

Listing 4: Zatrzymanie serwera

[3]jeśli zostało to skonfigurowane zgodnie z `https://dev.mysql.com/doc/refman/8.0/en/server-logs.html`

[UWAGA!] *Użytkownicy w systemie MySQL są całkowicie niezależni od użytkowników systemu operacyjnego Microsoft Windows czy UNIX/LINUX, chyba że zostało to skonfigurowane inaczej* [4].

1.5 Diagnostyka serwera MySQL z wykorzystaniem narzędzia mysqladmin.exe i klienta mysqlshow.exe

Mysqladmin jest narzędziem przeznaczonym do wykonywania operacji administracyjnych w obrębie SZDB. Jego głównym przeznaczeniem jest kontrola konfiguracji i aktualnego stanu serwera, tworzenia i usuwania baz danych itp.

Dokumentacja ! Przeczytaj i zapoznaj się z pełną dokumentacją
`https://dev.mysql.com/doc/refman/8.0/en/mysqladmin.html`

[Uwaga!]. Polecenie wykonaj z poziomu katalogu mysql "basedir"

```
    bin\mysqladmin -u root -p variables

    bin\mysqladmin -u root -p version

    bin\mysqladmin -u root -p processlist

    bin\mysqladmin -u root -p kill id,id....

    bin\mysqladmin -u root -p create db_name

```

Listing 5: Diagnostyka serwera

Klienta[5] **mysqlshow.exe** służy do szybkiej kontroli i przeglądu bazy danych, ich tabele, kolumny czy indeksów tabel. Mysqlshow zapewnia interfejs wiersza poleceń dla SQL-owej instrukcji SHOW.

[Uwaga!]. Polecenie wykonaj z poziomu katalogu mysql "basedir"

```
    bin\mysqlshow -u root -p

    bin\mysqlshow -u root -p mysql

    bin\mysqlshow -u root -p mysql user

```

Listing 6: Wyświetlenie informacji o bazie danych

[5] narzędzie

1.6 Uruchomienie klienta mysql

Należy pamiętać, że serwer bazodanowy MySQL oparty został o architekturę klient-serwer[6]. Mimo tego, iż fakt ten jest ogólnie znany, intuicyjny i niezaskakujący, to nader często szczególnie w przypadku młodych użytkowników tej technologii, dochodzi do próby łączenia się z usługą która nie została wcześniej uruchomiona (czytaj *mysqld.exe*) jak i mylnego utożsamiania klienta serwera MySQL (mysql.exe) z samym serwerem bazodanowy (mysqld.exe).

Mysql.exe to prosta powłoka SQL z możliwością edycji i wprowadzania poleceń w języku SQL. W przypadku użycia trybu interaktywnego wyniki zapytania są prezentowane w formacie tabeli ASCII. W trybie nieinteraktywnym (na przykład jako filtr), wynik jest prezentowany w formacie oddzielonym tabulatorami. Format wyjściowy można zmienić za pomocą opcji poleceń.

```
1    bin\mysql -u root -p -h localhost -P 3306
```

Listing 7: Uruchomienie klienta mysql

Mysql wysyła każdą wydaną instrukcję SQL do serwera w celu wykonania. Istnieje również zestaw poleceń, które sam mysql[7] interpretuje. Aby uzyskać listę tych poleceń, wpisz help lub \h po znaku zachęty serwera (*mysql>*).

[Uwaga!]. Polecenie wykonaj z poziomu klienta mysql, po wcześniejszym zalogowaniu. Zapoznaj się z możliwościami klienta mysql.

```
1    mysql> help
```

Listing 8: Uruchomienie klienta mysql

1.7 Polecenie SHOW (Database Administration Statements)

Podstawowym zadaniem klauzuli **SHOW**[8] jest dostarczenie informacji o bazach danych, tabelach, kolumnach lub informacji o statusie serwera bazodanowego.

Dokumentacja ! Poniżej zaprezentowano jedynie kilka podstawowych poleceń, całą listę dostępnych kombinacji znajdziesz w dokumentacji.
`https://dev.mysql.com/doc/refman/8.0/en/show.html`

[6] Podobnie jak ORACLE, POSTGRES, MSSQL i wiele innych
[7] klient, mysql.exe
[8] wywołanej jako polecenie języka SQL

Poniżej wykonaj i sprawdź działanie kilku zapytań z zakresu administrowania bazą danych.

```
mysql> SHOW DATABASES;
```

Listing 9: Listowanie dostępnych baz

```
mysql> SHOW TABLES;
```

Listing 10: Listowanie tabel domyślnej bazy / konieczne jest wybranie bazy domyślnej

```
mysql> USE db_name;
```

Listing 11: Wybieranie domyślnej bazy

```
mysql> SHOW TABLES;
```

Listing 12: Wyświetlenie listy tabel

```
mysql> \h SHOW
```

Listing 13: Lista możliwości polecenia show

Instrukcja **DESCRIBE** jest synonimem, używanym do uzyskiwania informacji o strukturze tabel. W przypadku tabeli zadziała dokładnie jak 'SHOW COLUMNS FROM'

```
mysql> DESC user;

lub

mysql> DESCRIBE user;

lub

mysql> SHOW COLUMNS FROM user;
```

Listing 14: Wyświetlenie struktury tabeli user

2 DQL - Podstawy

Język *'zapytań o dane'* (**DQL**) jest jednym z najistotniejszych elementów podziału pod języków SQL. Podgrupy te dzielą język SQL na cztery głównie kategorie: język zapytań o dane (**DQL**), język definicji danych (**DDL**), język kontroli danych (**DCL**) i język manipulacji danymi (**DML**).

2.1 Zmienne systemowe

Klauzula **SET** ma kilka form i różnorakie zastosowanie. Jednakże, na potrzeby niniejszego rozdziału, najbardziej nas umożliwienie przypisywania wartości do zmiennych. Poniżej przedstawiono kwerendę przypisującą łańcuch znaków do zmiennej sesyjnej.

```
1
2     mysql> SET @imie = 'Janek';
3
```

Listing 15: Deklarowanie i przypisywanie wartości do zmiennej

Zmienne zdefiniowane przez użytkownika, są specyficzne dla sesji. Ponadto, zmienne sesyjne nazywane również zmiennymi użytkownika są hermetyczne, to znaczy zdefiniowane przez jednego klienta nie mogą być widziana/wykorzystane przez innych. Wyjątek stanowi użytkownik z dostępem do tabeli schematu o nazwie *'user_variables_by_thread'*, może on przeglądać wszystkie zmienne użytkowników niezależnie od sesji.

Wszystkie zmienne danej sesji są automatycznie zwalniane, gdy klient **kończy działanie**. W nazwach zmiennych użytkownika nie jest rozróżniana wielkość liter, a nazwy mogą przyjąć **maksymalną długość 64 znaków**. Mimo tego iż opis tego polecenia znajduje się w rozdziale DQL, klauzula SET nie jest częścią te podgrupy a należy do bazowych instrukcji języka.

```
1     mysql> SELECT @imie;
```

Listing 16: Selekcja/projekcja zmiennej

Pamiętaj, operator przypisania '=' jest traktowany jako operator porównania w instrukcjach innych niż SET, dobrym nawykiem jest używania operatora ':='.

```
1     mysql> SET @imie = 'Janek';
2     lub
3     mysql> SET @imie := 'Marek';
```

Listing 17: Operatory przypisania dla SET

UWAGA ! Zmiennym użytkownika można przypisać wartość z ograniczonego zestawu typów danych: **liczby całkowite, dziesiętne, zmiennoprzecinkowe, łańcuchy binarne, niebinarne lub wartości NULL**. Przypisanie wartości dziesiętnych i rzeczywistych nie zachowuje precyzji ani skali wartości!

Wartość innego typu, są konwertowane do postaci typu dozwolonego! np. wartość o typie danych czasowych lub przestrzennych są konwertowane na ciąg binarne.

`https://dev.mysql.com/doc/refman/8.0/en/user-variables.html`

Zmienne użytkownika mają na celu dostarczanie wartości danych. Nie można ich używać bezpośrednio w instrukcji SQL jako identyfikatora lub jego części. Na przykłada w kontekstach, w których oczekiwana jest nazwa tabeli lub bazy danych, jak również jako słowa zastrzeżonego np. SELECT. Dzieje się tak nawet wtedy, gdy zmienna jest cytowana. Wyjątkiem od tej zasady, jest sytuacja, gdy konstruujemy ciąg jako "instrukcja sql do późniejszego wykonania". W takim przypadku zmienne użytkownika mogą być użyte do dostarczenia dowolnej części instrukcji. Poniższy przykład ilustruje, jak można to zrobić.

```
1    mysql> SET @column_name = "user";
2    mysql> SET @s = CONCAT("SELECT ", @column_name, " FROM user");
3    mysql> PREPARE stmt FROM @s;
4    mysql> EXECUTE stmt;
5
6    mysql> DEALLOCATE PREPARE stmt;
```

Listing 18: Pobranie listy nazw użytkowników serwera MySQL

```
1    mysql> SET @skip=1; SET @numrows=2;
2    mysql> PREPARE STMT FROM 'SELECT * FROM user LIMIT ?, ?';
3    mysql> EXECUTE STMT USING @skip, @numrows;
4
5    mysql> DEALLOCATE PREPARE stmt;
```

Listing 19: Pobranie limitowanej listy nazw użytkowników serwera MySQL

Dokumentacja ! Bardziej dokładny opis przygotowania kwerend znajduje się w dokumentacji
`https://dev.mysql.com/doc/refman/8.0/en/sql-prepared-statements.html`

2.2 Import danych

Na potrzeby ćwiczeń pobierz repozytorium zawierające zrzut demonstracyjnej bazy danych przygotowany przez producenta oprogramowania, znajdziesz go pod linkiem `https://github.com/datacharmer/test_db`.

Konieczne jest jego rozpakowanie i umieszczenie zawartości w katalogu *'employees'*, który należy utworzyć w katalogu bazowym (sprawdź ustawienie ścieżki 'basedir' w pliku my.ini).

```
> mkdir employees
> cd employees
```

Listing 20: wykonaj z poziomu katalogu bazowego

[UWAGA!] Importu dokonaj z poziomu katalogu employees.

```
..\bin\mysql -t < employees.sql
```

Listing 21: Import danych za pomocą strumienia systemowego

```
..\bin\mysql -t < test_employees_sha.sql
```

Listing 22: Walidacja poprawności danych

Jak można było zauważyć, dzięki raportowemu podsumowaniu, zaimportowana baza posiada znaczącą ilość rekordów, jak i wszystkie możliwe relacje zachodzące pomiędzy encjami (prócz odwołanie się do samej siebie). Dając nam tym samym szerokie możliwości w kontekście realizacji nawet skomplikowanych zapytań testowych

2.3 Selekcja i projekcja

Dokumentacja ! Przeczytaj i zapoznaj się z dokumentacją `https://dev.mysql.com/doc/refman/8.0/en/select.html`

Pomimo iż intuicyjne rozróżnienie operatorów algebry relacyjnej jakimi są selekcja i projekcja nie jest niczym trudnym, tak już określenie kolejności wykonywanych działań i zależności w obrębie złożonego zapytania sql, nie jest już tak oczywiste. Konieczne jest dokonanie omówienia i przedstawienie na przykładach zależności i procesów realizowanych w obrębie SZDB, w oparciu o zapytania z podgrupy DQL, aby proces ten ułatwić.

```
mysql> SELECT * FROM user;
lub
mysql> SELECT user,host,autentication_string FROM user;
```

Listing 23: Pobranie wszystkich/wybranych wartości z tabeli user

Dokonaj wybrania danych pracowników z tabeli employees pobierając jedynie drugą dziesiątkę pracowników. Sprawdź jak działa limit i jak deklarowany jest offset.

```
mysql> SELECT * FROM employees LIMIT 10,10;
```

Listing 24: Limitowanie wynikó

2.3.1 Wybrane operatory

Dokumentacja ! Przeczytaj i zapoznaj się z dokumentacją
`https://dev.mysql.com/doc/refman/8.0/en/non-typed-operators.html`

Wykonaj polecenia i przeanalizuj wyniki selekcji. (Rzutując na wszystkie atrybuty)

```
mysql> SELECT * FROM employees WHERE emp_no > 10100 LIMIT 10;
mysql> SELECT * FROM employees WHERE emp_no < 10100 LIMIT 10;
mysql> SELECT * FROM employees WHERE emp_no >= 10100 LIMIT 10;
mysql> SELECT * FROM employees WHERE emp_no != 10100 LIMIT 10;
```

Listing 25: Wybrane operatory porónania

```
mysql> SELECT * FROM employees WHERE emp_no < 10100 OR
            first_name != 'Tom' LIMIT 1000;

mysql> SELECT * FROM employees WHERE birth_date > '1960-01-01'
            AND emp_no > 11000 LIMIT 10;

mysql> SELECT * FROM employees WHERE emp_no
            NOT IN (10001,10002,10003) LIMIT 10;
```

Listing 26: OR/AND/NOT

```
mysql> SELECT * FROM employees WHERE emp_no BETWEEN 10000 AND
20000;
```

Listing 27: BEETWEN

```
mysql> SELECT * FROM employees WHERE first_name LIKE 'Smith';
mysql> SELECT * FROM employees WHERE first_name LIKE 'S%h';
mysql> SELECT * FROM employees WHERE first_name LIKE 'S%';
mysql> SELECT * FROM employees WHERE first_name LIKE 'S_ith';
```

Listing 28: LIKE

Wykonaj polecenia i przeanalizuj wyniki projekcji. (Selekcja realizowana na podstawie operatorów i przykładów powyżej)

Pamiętaj, iż nie wszystkie polecenia mogą wykonać się (**!!!**) lub być zadeklarowane poprawnie. **Istotą zadania jest analiza.**

```
mysql> SELECT empl_no, first_name, last_name, birth_date
    FROM employees LIMIT 10;

mysql> SELECT empl_no numer, first_name imie, last_name nazwisko,
    birth_date data_urodzenia
    FROM employees WHERE first_name LIKE 'T%' LIMIT 10;

mysql> SELECT e.empl_no numer, e.first_name AS imie,
    e.last_name nazwisko, e.birth_date data_urodzenia
    FROM employees e FROM WHERE imie LIKE 'T%' LIMIT 10;

mysql> SELECT `employees`.`empl_no numer`,
    `employees`.`first_name` AS imie,
    `employees`.`last_name` nazwisko,
    `employees`.`birth_date` data_urodzenia
    FROM `employees` FROM WHERE
    `employees`.`first_name` LIKE 'T%' employees LIMIT 10;

mysql> SELECT e.empl_no numer, e.first_name AS imie,
    e.last_name nazwisko, e.birth_date data_urodzenia
    FROM employees e FROM WHERE e.first_name
    LIKE 'T%' ORDER BY e.first_name ASC, e.last_name
    DESC LIMIT 1000;

 mysql> SELECT e.empl_no numer, e.first_name AS imie,
    e.last_name nazwisko, e.birth_date data_urodzenia
    FROM employees e FROM WHERE e.emp_no > 11000
    ORDER BY imie ASC, e.last_name DESC LIMIT 1000;

```

Listing 29: Projekcja / aliasy / grawisy / sortowanie

Począwszy od zapytania drugiego, wprowadzono do zapytań instytucję aliasów, inaczej mówiąc 'skrótów roboczych' możliwych do użycia w przypadku kolumn, tabel czy podzapytań. Obecnie rozróżniamy starą i nową notację. Różnią się one użyciem słowa kluczowego AS. Aktualnie preferuje się pomijanie słowa AS przed nazwą aliasu.

Ponadto, w przedstawionym listingu zauważyć można użycie znaku specjalnego '.' i '`'. Znak kropki odpowiada za zagłębianie struktury zbiorów lub obiektów [9]. Znak '`' określany jest mianem *'grawisu'*, służy on do "wescapowania"(nieinterpretowania) słów kluczowych i spacje w nazwach tabel i kolumn.

Istotną częścią zapytania piątego powyższego listingu jest klauzula ORDER BY, umożliwia ona sortowanie wyników selekcji. Zauważ iż może on zostać przeprowadzona w oparciu o aliasy, co jasno wskazuje na kolejność wykonywanych działań przez optymalizator i silnik SZDB.

[9] np bazy danych, tabeli, kolumny. Często instytucja zagłębienia jest wykorzystywany, gdy domyślna baza danych nie została wybrana lub gdy dokonywane jest złączenie pomiędzy tabelami z różnych baz

Opisując istotne elementy języka zapytań nie możemy zapomnieć o znaku zatwierdzenia[10] rozkazu, nosi on miano DELIMITERA. Domyślnie większość systemów bazodanowych przyjmuje, jako znak końca rozkazu średnik. Wartość ta może zostać zmieniona, choć dobrą praktyką jest pozostawienie domyślnej wartości. Istnieją jednak odstępstwa od tej Reguły np. podczas deklarowania procedur.

```
mysql> DELIMITER $$
mysql> SELECT * FROM employees WHERE emp_no > 10100 LIMIT 10$$
mysql> DELIMITER ;
mysql> SELECT * FROM employees WHERE emp_no > 10100 LIMIT 10;
```

Listing 30: Zmiana delimitera

2.3.2 Funkcje przepływu

Dokumentacja ! Przeczytaj i zapoznaj się z dokumentacją
`https://dev.mysql.com/doc/refman/8.0/en/flow-control-functions.html`

Funkcje sterowania przepływem, to grupa funkcji pozwalających na ocenę wartości atrybutu. Do grupy tej należą CASE -operator przypadku, IF() klasyczna konstrukcja if/else, IFNULL() konstrukcja null if/else, NULLIF() konstrukcja która zwróć NULL, jeśli wyrażenie1 = wyrażenie2. [**Pamiętaj!**] iż funkcje przepływu to nie instrukcje warunkowe, te poznamy podczas deklarowania procedur i funkcji.

```
mysql> SELECT empl_no, first_name, last_name,
         IF(YEAR(birth_date) < 1970, 'starzec', 'mlodzian)'
     FROM employees LIMIT 10;

```

Listing 31: Funkcja sterowania przepływem - IF

```
mysql> SELECT CASE
            WHEN first_name = 'Tom' THEN 'Tomek'
            WHEN first_name = 'John' THEN 'Janek'
          END imie_po_ocenie,
          first_name orginalne_imie,
     FROM employees;
```

Listing 32: Funkcja sterowania przepływem - CASE

2.3.3 Funkcje daty/czasu, funkcje łańcuchowe, funkcje numeryczne

Dokumentacja ! Przeczytaj i zapoznaj się z dokumentacją
`https://dev.mysql.com/doc/refman/8.0/en/functions.html`

[10] zakończenia

Poniżej przestawiono listę funkcji z jakimi szczególnie należ się zapoznać w kontekście remizowych zadań: ADDDATE(), ADDTIME(), DATE(), DAY(), MONTH(), YEAR(), DAYOFWEEK() ,TIMESTAMP, NOW() UNIX_TIMESTAMP(), UPPER(), LOWER(), LEFT(), RIGHT(), SUBSTR(), LENGHT(), LTRIM(), FIND_IN_SET(), LIKE(), POW(), RAND(), SIN(), COS(), FLOOR(), PI(), DIV(), CONV(),

2.4 Ładowanie i eksport danych z pliku csv/txt

Dokumentacja ! Przeczytaj i zapoznaj się z dokumentacją
`https://dev.mysql.com/doc/refman/8.0/en/select-into.html`
`https://dev.mysql.com/doc/refman/8.0/en/load-data.html`

Przed konfiguracją serwera zezwalającą na ładowanie i eksport danych za pośrednictwem klauzul języka sql z i do katalogu, sprawdź wartość zmiennej globalnej 'secure_file_priv'.

```
1    SELECT @@global.secure_file_priv;
```

Listing 33: Weryfikacja uprawnień odczytu/zapisu

Import danych wymaga zatrzymania serwera bazy danych i wprowadzania do pliku konfiguracyjnego odpowiednich parametrów, zezwalających na odczyt danych z ustalonego katalogu. Istnieje również możliwość konfiguracji i wprowadzania koniecznych zmian z poziomu języka sql, odwołując się do zmiennej globalnej lub uruchomienia serwer w specjalnym trybie. Funkcjonowanie poszczególnych scenariuszy, uzależnione jest od wersji serwera, uprawnień użytkownika bazy danych jak i uprawnień systemowych. [**Pamiętaj !**] Nie stosuj scenariuszy razem.

```
1    bin\mysqld --secure-file-priv=""
```

Listing 34: Scenariusz 1. Uruchomienie serwera w trybie secure-file-priv

```
1    [mysqld]
2    # set basedir to your installation path
3    basedir=C:\ISNS\mysql
4    # set datadir to the location of your data directory
5    datadir=C:\ISNS\mysql\data
6    secure-file-priv=""
```

Listing 35: Scenariusz 2. Wprowadzanie zmian do pliku konfiguracji serwera

```
1    SELECT @@global.secure_file_priv;
```

Listing 36: Weryfikacja uprawnień odczytu/zapisu

Kwerenda eksportu danych do pliku przedstawiona poniżej, realizuje przygotowanie wszystkich rekordów z tabeli *'employees'* oraz projekcję wykonaną na wszystkich atrybutach zapewnioną poprzez *'wildcard'* * , a następnie przekazanie ich do pliku zewnętrznego. [**Pamiętaj !**]Bez podania ścieżki plik powstanie w katalogu bazy)

```
SELECT * INTO OUTFILE 'C:\\ISNS\\mysql\\employees.csv'
FIELDS TERMINATED BY ';' OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\r\n'
FROM employees;
```

Listing 37: Eksport danych z tabeli employees

Dokonaj przeliczenia rekordów tabeli.

```
mysql> SELECT COUNT(*) FROM employees2;
```

Listing 38: Przeliczenie ilości wierszy tabeli

Za pomocą edytora dokonaj edycji pliku *.csv* dodając do pliku dziesięciu pracowników nadając im *'emp_no'* od numeru 80 do 89.

```
CREATE TABLE employees2 LIKE employees
```

Listing 39: Tworzenie tabeli na podstawie struktury innej tabeli

Zaimportuj pliki i sprawdź wynik

```
LOAD DATA INFILE 'C:\\ISNS\\mysql\\employees.csv' INTO TABLE
employees2
FIELDS TERMINATED BY ';' OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\r\n';
```

Listing 40: Import danych do tabeli employees2

Dokonaj ponownego przeliczenia i sprawdzenia wyniku działań.

```
mysql> SELECT * FROM employees2 WHERE empl_no BETWEEN 80 AND 89;

mysql> SELECT COUNT(*) FROM employees2;
```

Listing 41: Kontrola danych

2.5 Podzapytania

Ze względu na rozlokowanie danych zgodnie z zasadami normalizacji danych, gdzie informacje rozdzielane są na poszczególne encje, wyłuskanie danych może wymagać od nas pozyskania ich z innych tabel czy pojedynczych rekordów. Podzapytanie jest poleceniem SELECT zagnieżdżonym w innym poleceniu SELECT. Podzapytanie może wystąpić wszędzie tam, gdzie system spodziewa się zbioru wartości, między innymi w klauzulach SELECT, FROM, WHERE, HAVING lub literału(zmiennej), gdy podzapytanie zawraca pojedynczą wartość.

2.5.1 Podzapytania tablicowe

Podzapytanie tablicowe to zapytanie które zwraca zbiór rekordów[11]. Podzapytania tablicowe dopuszczają zastosowanie operatorów: IN, ANY, ALL

```
mysql> SELECT empl_no, first_name, last_name
        FROM employees WHERE emp_no IN
        ( SELECT emp_no FROM salaries WHERE salary > 70000)
```

Listing 42: Podzapytanie nieskorelowane

2.5.2 Podzapytania wierszowe

Podzapytania wierszowe zwraca zawsze jedną krotkę, w obrębie której znajduje się jedna lub wiele wartości (lista). Podzapytanie to dopuszcza umożliwia wykorzystanie następujących operatorów logicznych: =, !=, >, <>, <, <=, >=.

```
mysql> SELECT empl_no, first_name, last_name
        FROM ( SELECT * FROM employees WHERE emp_no%2 = 0)
```

Listing 43: Podzapytanie nieskorelowane

2.5.3 Podzapytania nieskorelowane

Podzapytania nieskorelowane inaczej mówiąc niepowiązane to zapytanie niezależne od zapytania zewnętrznego i może być wykonane samodzielnie.

```
mysql> SELECT empl_no, first_name, last_name
        FROM ( SELECT * FROM employees WHERE emp_no%2 = 0)
```

Listing 44: Podzapytanie nieskorelowane

```
mysql> SELECT empl_no, first_name, last_name  FROM employees
        WHERE emp_no IN
(SELECT DISTINCT emp_no FROM salaries WHERE salary > 60000)

```

Listing 45: Podzapytanie nieskorelowane 2

DISTINCT - klauzula ta pozwala na usunięcie powtórzeń rekordów o tych samych wartościach.

[11] zbór może być jednoelementowy

2.5.4 Podzapytania skorelowane

Podzapytania skorelowane inaczej mówiąc powiązane to zapytanie zależne od zapytania zewnętrznego i nie może być wykonane samodzielnie.
Poniższy przykład przedstawia podzapytanie które na podstawie informacji z zapytania zewnętrznego a dokładnie wartości e.emp_no, dokonując selekcji w tabeli salaries i zwracając wartość sumy uposażenia (funkcja agregująca), a próba jego uruchomienia poza zapytaniem głównym zakończy się niepowodzeniem.

```
mysql> SELECT empl_no, first_name, last_name,
    (SELECT SUM(salary) FROM salaries s
    WHERE s.emp_no = e.emp_no ) wuma_wyplat
    FROM employees e;
```

Listing 46: Podzapytanie skorelowane

2.6 Zadania

(a) Wybierz wszystkich pracowników, których nazwiska zaczynają się na literę J, ustawiając limit wierszy na 100.

(b) Wybierz wszystkich pracowników, których imię zaczyna się od litery 'A' i kończy na literze 'a', ustalając limit wierszy na 100.

(c) Wybierz wszystkich pracowników, których imię zaczyna się od litery 'T' i trzecia litera imienia to 'n', ustalając limit wierszy na 100.

(d) Dokonaj zamiany pierwszej litery nazwiska na 'A', których imię zaczyna się od litery 'T' i trzecia litera imienia to 'n', ustalając limit wierszy na 100.

(e) Dokonaj wybrania danych na temat imienia, nazwiska i miesiąca urodzenia pisanego słownie dla drugiej setki pracowników.

(f) Dokonaj wybrania danych na temat imienia, nazwiska i miesiąca urodzenia pisanego słownie dla pracowników o emp_no = 10001, 10002, 10004.

(g) Dokonaj wybrania danych na temat imienia, nazwiska i miesiąca urodzenia pisanego słownie dla drugiej setki pracowników. Oceń i w kolumnie 'ocena' umieść znacznik 'starzec' lub 'młodzian' gdy rok urodzenia osoby jest mniejszy od 1975

(h) Wybierz pierwszy okres rozliczeniowy dla pracownika, podając datę rozpoczęcia pracy. Projekcja powinna zawierać imię, nazwisko, datę początku okresu rozliczeniowego pierwszej wypłaty.

(i) Wybierz ostatni okres rozliczeniowy dla pracownika, podając datę zakończenia pracy. Projekcja powinna zawierać imię, nazwisko, datę końca okresu rozliczeniowego pierwszej wypłaty.

(j) Pobierz imię i nazwisko wszystkich osób, których suma wypłat przekroczyła 200 tysięcy. W projekcji dokonaj przedstawienia imienia, nazwiska, wartości wypłaty.

3 DQL - Złączenie

Złączenie to element algebry relacyjnej odpowiadający za połączenia krotek (danych) rezydujących w dwóch lub więcej tabelach z wykorzystaniem optymalnej metody/algorytmu.

Istnieje kilka rodzajów połączeń danych z kolokowanych w tabelach,:

- iloczyn kartezjański
- oraz złączenia (naturalne, równościowe, nierównościowe, zewnętrzne i zwrotne).

Ich omówienia dokonamy poniżej.

3.1 Iloczyn kartezjański - CROSS JOIN

Najprostszy typ połączenia, często opisywany za pomocą wyrażenia CORSS JOIN. Wynikiem połączenia tabel z wykorzystaniem iloczynu kartezjańskiego będzie tabela, zawierająca wszystkie atrybuty z obu tabel, zaś krotki będą stanowić iloczyn rekordów oryginalnych tabeli (przy założeniu, w którym brak jest ograniczeń lub warunków selekcji). Wykorzystanie iloczynu kartezjańskiego w obszarze rozleglejszych relacji (tabel) potrafi w sposób znaczący obciążyć SZBD, a nawet doprowadzić do wystąpienia i zwrócenia przez SZDM błędu zapytania. Obecnie iloczyn kartezjański w czystej postaci rzadko bywa przydatny, a konieczność jego użycia należy każdorazowo przemyśleć.

```
1    SELECT ...
2    FROM <relacja> [CROSS JOIN] <relacja>
3    ...
4    [WHERE ...]
5    [ORDER BY ...]
6    <relacja> = { table|view|cte| ( select stmt )}
7                [[AS] alias]
```

Listing 47: CROSS JOIN

Zadanie 1.. Dokonaj połączenia tabel employees i salaries z wykorzystaniem CROSS JOIN

—Warto zapamiętać—

```
1    mysql> SELECT * FROM information_schema.processlist WHERE user =
     ' root ';
2
3    mysql> KILL id_procesu ;
```

Listing 48: Listowanie i zatrzymywanie uruchomionego procesu bez restartu SZDB.

Zadanie 2. Zlokalizuj i zatrzymaj proces uruchomiony w poprzednim zadaniu bez zatrzymywania SZDB.

3.2 Połączenia równościowe - JOIN

Wynikiem poprzednio wykonywanego zadania będzie lista wszystkich możliwych kombinacji krotek pochodzących z tabeli employees i salaries. Jednakże, jak zapewne już zauważyliście, tabele te posiadają atrybut umożliwiający jednoznaczną identyfikację pracownika. Atrybut ten to nim 'emp_no', pozwoli on dokonać odpowiedniego dopasowania i wyekstrahowanie znacznie istotniejszej dla nas informacji, to znaczy przypisania do każdej zrealizowanej wypłaty danych osobowych odbiorcy.

Tego rodzaju działanie wykonywać będziemy za pomocą złączenia typu [INNER] JOIN, a konstrukcja tabeli wynikowej i rekordów znajdujących w niej będzie wypadkową dopasowania rekordów z tabeli employees i salaries, zrealizowaną na podstawie zadanego przez nas warunku. Istotnym jest, aby atrybuty tworzące warunek pochodziły bezpośrednio z łączonych tabel, dodatkowo mile widzianą praktyką jest używanie do tego celu pary kluczy *'Tabela A - primary key'->'Tabela B foreign key z A'*.

```
1    SELECT alias1.atrybutN, alias2.atrybutM ........
2    FROM tabela1 [alias1] [INNER] JOIN tabela2 [alias2]
3        ON warunek polaczenia
4    [WHERE ....]
5    [ORDER BY .....]
```

Listing 49: JOIN

Pamiętać należy, aby warunek połączenia realizowany był z wykorzystaniem znaku '=' zapewniając nam wybór i dopasowanie jedynie rekordów posiadających parę (złączenie równościowe w którym wybrano operator '=' oraz deklarację JOIN, wybiera część wspólną dwóch zbiorów niejawnie deklarujące INNER JOIN). Ponadto w przypadku wystąpienia w obu tabelach kolumny o tej samej nazwie, a w szczególności w sytuacji wykorzystania ich do stworzenia warunku złączenia, zobligowani jesteśmy do użycia klasyfikatora obiektu przed nazwą atrybutu lub jego aliasu np. employees.emp_no | em.emp_no.

W podobny sposób należy poprzedzać nazwy atrybutów (aliasami albo nazwami tabel) w wy- rażeniach w klauzulach: SELECT , WHERE , albo ORDER BY .

Zadanie 3. Wybierz i dopasuj do wypłaty użytkownika. Projekcja ma przedstawić wszystkie atrybuty pochodzące z obu tabel.

Zadanie 4. Wybierz i dopasuj do wypłaty użytkownika. Projekcja ma przedstawić jedynie imię i nazwisko, kwotę, okres od, okres do oraz identyfikator pracownika.

Zadanie 5. Wybierz i dopasuj do użytkownika informacje na temat wypłaty, pod warunkiem, iż kwota wypłaty przekroczyła 10 000, a nazwisko pracownika zaczyna

się na literę 'A'. Projekcja ma przedstawić jedynie imię i nazwisko, kwotę, okres od, okres do oraz identyfikator pracownika.

3.3 Złączenie naturalne - NATURAL JOIN

Złączenie naturalne jest specyficznym rodzajem złączeń równościowych. Złączenie naturalne tabel to połączenie w którym warunki równości dotyczą wszystkich par atrybutów o takich samych nazwach. "Podstawową różnicą, pomiędzy zapytaniami równościowymi, a naturalnymi, jest lista atrybutów tabel powstającej w wyniku połączenia. W wyniku połączenia naturalnego atrybut (albo atrybuty) połączeniowe występują tylko raz, podczas gdy w wyniku połączenia równościowego występują oba atrybuty połączeniowe z obu łączonych tabel."[12]

```
1    SELECT ...
2    FROM <tabela> NATURAL [<join type>] JOIN <tabela>
3    ...
4    <relacja> = { table|view|cte|(select stmt)[[AS] alias]}
5    <join type > = INNER | {LEFT|RIGHT|FULL} [OUTER]
```

Listing 50: NATURAL JOIN notacja 1:

```
1    SELECT ...
2    FROM <tabela> [<join type>] JOIN <tabela>
3    USING ( col_name [, col_name ... ] )
```

Listing 51: NATURAL JOIN notacja 2:

Różnice pomiędzy obiema notacjami są dość istotne. W notacji pierwsze atrybuty o tych samych nazwa muszą być równe a ich ilość określa SZDB, w notacji drugiej zaś możemy określić które kolumny będą podlegać porównaniu. W przeciwieństwie jednak do złączeń równościowych, za deklaracją drugiej tabeli(relacji) użyto operatora **USING** a nie **ON** i podano listę atrybutów stanowiącą wypadkową cześć wspólną obu tabel, które to mają zostać wykorzystane do połączenia. Stanowi to znaczącą różnicę.

Zadanie 6. Dokonaj połączenia tabel employees i salaries z wykorzystaniem NATURAL JOIN, z wykorzystanie obu notacji.

3.4 Złączenia nierównościowe

Złączenia nierównościowe są połączeniami, w których warunek połączeniowy nie korzysta z operatora równości, a do działania wykorzystuje inne operator (np. BETWEEN, IN itp). Podobnie jak w przypadku połączenia równościowego, w wy-

[12]'BAZY DANYCH — lab' - dr inż. Antoni Ligęza 2009

niku połączenia nierównościowego powstaje relacja (tabela), która zawiera wszystkie atrybuty z obu tabel."Znajdowane są wszystkie pary krotek, z których jedna pochodzi z pierwszej łączonej tabeli, a druga z drugiej i spełniają one warunki połączenia". [13]

```
mysql> SELECT  e.emp_no, e.first_name, e.last_name, d.emp_no
 FROM employees e JOIN dept_emp d ON e.emp_no
 BETWEEN (d.emp_no+10000) AND (d.emp_no+10003)
LIMIT 10;
```

Listing 52: Złączenie nierównościowe

3.5 Złączenia zewnętrzne [LEFT | RIGHT] JOIN

Poznane do tej pory złączenia definiowaliśmy jako złączenia „wewnętrzne" (INNER JOIN), ze względu na naturę ich warunku, która musiała być spełniona, mówiąc jaśniej, wybierane i dopasowywane były tylko rekordy, które spieniały określony przez nas warunek. Jednakże istnieje możliwość wyboru wszystkich krotek jednej z relacji (tabeli) i dopasowania do nich krotek drugiej nawet w przypadku niespełnienia warunku (są to relacje typu LEFT/RIGHT JOIN).

Ponadto w zależności od zastosowanego systemu SZDB istnieje możliwość pobrania i wyświetlenia pełnej listy rekordów zarówno z 'tabeli A' jak i 'tabeli B' za pomocą FULL OUTER JOIN. Brakujące krotki połączenia wypełniane są wirtualnym dopasowaniem atrybutów, których wartość jest nieoznaczona (NULL).

```
SELECT alias1.atrybutN, alias2.atrybutM ........
FROM tabela1 [alias1] LEFT | RIGHT  JOIN tabela2 [alias2]
        USING(atrybutN)
[WHERE ....]
[ORDER BY .....]
```

Listing 53: OUTER JOIN notacjan 1 z wykorzystaniem USING

```
SELECT alias1.atrybutN, alias2.atrybutM ........
FROM tabela1 [alias1] LEFT | RIGHT  JOIN tabela2 [alias2]
        ON alias1.atrybutN = alias2.atrybutN
[WHERE ....]
[ORDER BY .....]
```

Listing 54: OUTER JOIN notacjan 2 z wykorzystaniem ON i atrybutu wspólnego

Zadanie 7. Wybierz wszystkich pracowników o numerze emp_no pomiędzy 20058 a 20063 i połącz ich z pobranymi wynagrodzeniami z tabeli salaries. Następnie dokonaj usunięcia w tabeli salaries rekordów dla emp_no = 20058 (poprzez polecenie

[13] 'BAZY DANYCH — lab' - dr inż. Antoni Ligęza 2009

DELETE FROM salaries WHERE emp_no = 20058;). Ponownie dokonaj złączenia z pierwszej części zadania, co zaobserwowałeś.

Zadanie 8. Dokonaj usunięcia użytkownika o numerze 20059 (z tabeli employees), a następnie wybierz wszystkie wypłaty (salaries) dla użytkownika o id 20059 i dopasuj do niego informacje z tabeli employees. Co zauważyłeś.

3.6 Operator zbiorowy - UNION

Operator zbiorowy pozwala na łączenia zbiorów, każde z zapytań DQL wylicza zbiór rekordów, a UNION umożliwia połączenie ich w jeden. Dostępne operatory to: operator UNION , wyliczający sumę dwóch zbiorów i eliminujący powtórzenia (domyślnie użyto przełącznika DISTINCT) ze zbioru wynikowego, operator UNION ALL wyliczający sumę dwóch zbiorów jednak bez eliminacji powtórzeń.

```
1    SELECT  atrybut [,atrybut ...] FROM <tabelaA>
2        UNION [ALL | DISTINCT ]
3    SELECT  atrybut [,atrybut ...] FROM <tabelaB>
```

Listing 55: Operator zbiorowy UNION

Zadanie 9. Na podstawie powyższych informacji dokonaj złączenie odwzorowującego FULL OUTER JOIN usuwając część wspólną pomiędzy relacjami employees i salaries.

4 DQL - Grupowanie i widoki

4.1 Grupowanie

Grupowanie to działanie sql mające na celu wyodrębnienie podzbiorów przetwarzanej kolekcji, odznaczających się tymi samymi właściwościami. Realizacja grupowania w większości SZDB, realizowana jest poprzez klauzulę GROUP BY jak i klauzule modyfikatorów, takich jak HAVING czy WITH ROLLUP. Użycie funkcji agregującej w instrukcji nie zawierającej klauzuli GROUP BY jest równoznaczne z grupowaniem po wszystkich wierszach. Dodatkowo o ile nie określono inaczej, funkcje agregujące ignorują wartości NULL.

Filtrowanie grup realizowane jest poprzez klauzule HAVING która działa na całych grupach wierszy. Klauzula ta określana jest również mianem selekcji poziomej (grup wierszy). [14]

Dokumentacja ! Przeczytaj i zapoznaj się z dokumentacją
`https://dev.mysql.com/doc/refman/8.0/en/aggregate-functions-and-modifiers.html`

4.1.1 Funkcje agregujące

- **COUNT()** – zwraca liczbę wierszy otrzymanych w wyniku zapytania,
- **COUNT(DISTINCT)** - zwraca liczbę unikalnych wartości (wierszy),
- **MIN()** – określa wartość minimalną dla wybranej kolumny w wyniku realizacji zapytania,
- **MAX()** – określa wartość maksymalną dla kolumny w w wyniku realizacji zapytania,
- **SUM()** – sumuje zawartość kolumny (lub wyrażenia) dla każdego wiersza będącego wynikiem zapytania,
- **AVG()** – wylicza średnią arytmetyczną zawartości kolumny (lub wyrażenia) dla każdego wiersza będącego wynikiem zapytania,
- **GROUP_CONCAT()** - funkcja zwraca wynik w postaci ciągu z połączonymi wartościami grupy innymi niż NULL. Umożliwiając obdzielenie ich poprzez określony znacznik np ",",
- **MEDIAN()** – wylicza medianę
- **STDDEV()** - oblicza odchylenie standardowe
- **VARIANCE()** - oblicza wariancję

Funkcje grupujące mogą zostać wykorzystane bez klauzuli GROUP BY, w takiej sytuacji cały zbiór traktowany jest jako pojedyncza grupa. Ponadto należy pamiętać, że wszystkie funkcje grupujące, z wyjątkiem funkcji COUNT(*), ignorują

[14] Analogicznie warunki określone w klauzurze WHERE, traktowane są jako selekcję poziomą pojedynczych rekordów.

wartości Null.

```
mysql> SELECT SUM(s.salary) suma, COUNT(*) ile
FROM employees e INNER JOIN salaries s ON e.emp_no = s.emp_no
WHERE e.emp_no BETWEEN 10000 AND 11000
```

Listing 56: Wykorzystanie funkcji agregujących

```
mysql> SELECT e.first_name, SUM(s.salary) suma, COUNT(*) ile
FROM employees e INNER JOIN salaries s ON e.emp_no = s.emp_no
WHERE e.emp_no BETWEEN 10000 AND 11000;
```

Listing 57: Wykorzystanie funkcji agregujących - zapytanie błędne

```
mysql>   SELECT e.first_name, e.last_name,
    SUM(s.salary) suma, COUNT(*) ile
FROM employees e INNER JOIN salaries s ON e.emp_no = s.emp_no
WHERE e.first_name = 'Sumant' AND e.last_name ='Peac';
```

Listing 58: Wykorzystanie funkcji agregujących 1

Zadanie 1 Dokonaj analizy powyższych listingów i przedstaw wnioski.

Zadanie 2 Sprawdź ustawienia swojego serwera za pomocą "SELECT @@session.sql_mode; ". Opisz za co odpowiada zmienna globalna ONLY_FULL_GROUP_BY ?

4.1.2 Grupowanie

Dokonaj przedstawienia listy wszystkich pracowników, podaj ich imię i nazwisko wraz z podsumowaniem zarobków za cały okres pracy. Wyniki posortuj po nazwisku i imieniu pracownika.

```
mysql> SELECT e.first_name, e.last_name, SUM(s.salary) suma
FROM employees e INNER JOIN salaries s ON e.emp_no = s.emp_no
WHERE e.emp_no BETWEEN 10000 AND 11000
GROUP BY e.first_name, e.last_name
ORDER BY e.last_name, e.first_name;
```

Listing 59: Grupowanie

Zadanie 3 Zmodyfikuj powyższe zapytanie zwracając informacje o ilości wypłaconych uposażeń, maksymalnej i minimalnej wartości uposażenia.

Zadanie 4 Na postawie informacji z zadania powyżej wylicz średnią i porównaj ją z wynikiem funkcji agregującej AVG().

Zadanie 5 Zbadaj działanie funkcji GROUP_CONCAT() i dokonaj zapisania w komunie "lista", listy wartości wszystkich uposażeń obdzielając wartości przecinkiem. Pamiętaj ilość elementów w liście musi pokrywać się z wartością zwracaną przez kolumnę "ilosc_wyplat"przeliczoną na podstawie funkcji COUNT(*). W projekcji wyświetl ponadto imię i nazwisko oraz sumę uposażeń otrzymanych przez danego pracownika.

4.1.3 Grupowanie z klauzulą HAVING

Dokonaj przedstawienia listy wszystkich pracowników, podaj ich imię i nazwisko wraz z podsumowaniem zarobków za cały okres pracy i ilością wypłat. Wyniki posortuj po nazwisku i imieniu pracownika przedstaw jedynie pracowników których suma wypłat przekroczyła 200 tyś a ilość była mniejsza od 5.

```
mysql> SELECT e.first_name, e.last_name,
SUM(s.salary) suma, COUNT(s.salary) ilosc_wyplat
FROM employees e INNER JOIN salaries s ON e.emp_no = s.emp_no
WHERE e.emp_no BETWEEN 10000 AND 11000
GROUP BY e.first_name, e.last_name
HAVING  SUM(s.salary) > 200000 AND COUNT(s.salary) < 5
ORDER BY e.last_name, e.first_name;
```

Listing 60: Grupowanie z klauzulą HAVING

Zadanie 6 Na podstawie powyższego listingu i wykorzystując funkcję HAVING, dokonaj odfiltrowania gdy średnia jednego uposażenia nie przekroczyła 45000.

4.1.4 Grupowanie z klauzulą WITH ROLLUP

Klauzula GROUP BY pozwala na użycie modyfikatora WITH ROLLUP, rozszerzającego wynik podsumowania o dodatkowe wiersze reprezentujące operacje podsumowujące wyższego poziomu (czyli super agregujące). ROLLUP w ten sposób umożliwia odpowiadanie na pytania na wielu poziomach analizy, wykorzystując do tego jedno zapytanie.

```
mysql> SELECT e.first_name, e.last_name, SUM(s.salary) suma,
COUNT(s.salary) ilosc_wyplat, t.title
FROM employees e
INNER JOIN salaries s ON e.emp_no = s.emp_no
INNER JOIN  titles t ON e.emp_no = t.emp_no
WHERE e.emp_no BETWEEN 10000 AND 10100
GROUP BY e.first_name, e.last_name, t.title ;
```

Listing 61: Grupowanie bez ROLLUP

```
mysql> SELECT e.first_name, e.last_name, SUM(s.salary) suma,
COUNT(s.salary) ilosc_wyplat, t.title
FROM employees e
INNER JOIN salaries s ON e.emp_no = s.emp_no
INNER JOIN  titles t ON e.emp_no = t.emp_no
WHERE e.emp_no BETWEEN 10000 AND 10100
GROUP BY e.first_name, e.last_name, t.title WITH ROLLUP;
```

Listing 62: Grupowanie z modyfikatorem ROLLUP

Zadanie 7 Dokonaj podsumowania uposażeń dla każdego pracownika, grupując dodatkowo zbiór po roku urodzenia i przedstawiając sumę uposażeń.

4.2 Widoki

"Widok jest predefiniowanym i zapisanym po stronie serwera zapytaniem, którego wynik może być wielokrotnie odczytywany. W standardzie ANSI SQL widoki określane są mianem tabel widokowych lub wirtualnych (ang. Viewed, virtual tables), natomiast tabele przechowujące dane noszą miano tabel bazowych (ang. Base tables).... [15]. Inną funkcjonującą w literaturze nazwą widoku, jest perspektywa.

Dokumentacja ! Przeczytaj i zapoznaj się z dokumentacją
`https://dev.mysql.com/doc/refman/8.0/en/create-view.html`
`https://dev.mysql.com/doc/refman/8.0/en/drop-view.html`

```
CREATE
    [OR REPLACE]
    [ALGORITHM = {UNDEFINED | MERGE | TEMPTABLE}]
    [DEFINER = user]
    [SQL SECURITY { DEFINER | INVOKER }]
    VIEW view_name [(column_list)]
    AS select_statement
    [WITH [CASCADED | LOCAL] CHECK OPTION]
```

Listing 63: Struktura instrukcji

```
    DROP VIEW [IF EXISTS]
    view_name [, view_name] ...
    [RESTRICT | CASCADE]
```

Listing 64: Usówanie widoków

Zadanie 8 Zbuduj widok w którym przedstawisz wszystkie kolumny z tabeli "employees"widok nazwij pracownicy. Sprawdź listę dostępnych tabel, zbadaj strukturę obiektu pracownicy poprzez polecenie DESCRIBE i SHOW CREATE VIEW analizując szczególnie polecenie SHOW.

[15] Poradnik webmastera - wydawnictwo Hellion - stan na 19.04.2023

Zadanie 9 Wybierz wszystkie atrybuty wcześniej stworzonego widoku dla użytkownika 10000 i 10001.

Zadanie 10 Wypełnij danymi rekord dla użytkownika o emp_no 10000 jako imię użyj "Jan"a nazwisko "Kowalski", resztę danych wypełnij według uznania. Ponów zapytanie z zadania 8.

Zadanie 11 Zmodyfikuj widok "pracownik", ograniczając atrybuty jedynie do first_name, last_name, emp_no, title.

Zadanie 12 Strwóż widok "pracownik2"w którym na etapie tworzenia zadeklarujesz nazwy kolumn widoku (poza poleceniem "SELECT". Nazwy kolumn widoku i ich dopasowanie powinno wyglądać następująco: imię -> first_name, nazwisko -> last_name, numer_pracownika -> empl_no.

Zadanie 13 Do wyżej stworzonego widoku spróbuj wprowadzić dane (nadaj emp_no wartość 9999) jak i usunąć jeden z rekordów (gdzie emp_no = 10003). CO zauważyłeś ?