Materiały do zajęć nr 7

Metody zapewnienia spójności danych

W SQL nie są stosowane automatyczne mechanizmy zapewnienia spójności danych, gdyż powodowałyby znaczące obniżenie szybkości przetwarzania danych. Zamiast tego dostępne są mechanizmy, które mogą być zastosowane stosownie do potrzeb określonych przez użytkownika.

Not null

Polega na nadaniu danej atrybutu **not null**, oznaczającego, że nie jest dopuszczalne wprowadzenie wartości null.

*Na potrzeby testów tworzę tabelę **T0***

Ustawienie atrybutu not null

```
create table T0(
      Id int,
      Name char(25) not null
);
desc T0;
```

```
insert T0 values (1024,'Nowak');
insert T0 (Id) values (1025);
```

```
=== ERROR ===
```

Komunikat wyświetlany do błędu jest w tym wypadku mylący, gdyż oznacza, że **Name** nie ma określonej wartości domyślnej. Tymczasem wartość domyślna jest określona, ale jest nią **Null**. Należy więc rozumieć, że nie ma wartości domyślnej innej niż **Null**.

Usunięcie atrybutu not null

```
alter table T0 modify Name char(15);

desc T0;
```

Czyli wystarczy wykonanie polecenia **alter modify**.

Default

Określa wartość domyślną, która jest wprowadzana, jeśli nic nie podamy.

*Na potrzeby testów tworzę tabelę **T0**.*

Ustawienie atrybutu default

```
drop table T0;
create table T0(
      Id int default 1,
      Name char(15) default ''
);

desc T0;

insert T0 values ();

select * from T0;
```

W tym przypadku wprowadzanymi danymi domyślnymi dla **Name** jest pusty łańcuch znakowy. To nie jest wartość **Null**!

Usunięcie atrybutu default

```
alter table T0 modify Name char(15);

desc T0;
```

Czyli wystarczy wykonanie polecenia **alter modify**.

Unique

Unique wprowadza atrybut unikalności, czyli dane nie mogą się powtarzać.

```
drop table T0, T1;
create table T0(Id int, Name char(25) unique);
create table T1(Id int, Name char(25) not null unique);
desc T0;              jest UNI
desc T1;              jest PRI


insert into T0 values (1024,'Nowak');
insert into T0 values (1025,'Nowak');

=== ERROR ===
```

Uwaga: atrybut UNIQUE nie działa w przypadku wartości pustych. Dlatego samo UNIQUE nie zapewnia jednoznacznej identyfikacji krotek:

```
insert T0 values ();

insert T0 values ();

select * from T0;
```

Tu błędu nie było, ale

```
insert T1 values ();

insert T1 values ();

=== ERROR ===
```

Unikalność zapewnia dopiero łączne użycie UNIQUE i NOT NULL. W MySQL odpowiada to atrybutowi klucza głównego (PRIMARY KEY).

Usunięcie atrybutu UNIQUE

```
drop table T0,T1,T2,T3;
create table T0(Id int, Name char(25));
alter table T0 modify Name char(25) unique;
desc T0;
alter table T0 modify Name char(25);
desc T0;                    nie działa
```

W przypadku UNIQUE nie wystarcza modyfikacja tabeli przez **alter modify**. Trzeba inaczej:

```
show index from T0;
alter table T0 drop index Name;          (może być też ...drop key ...)
desc T0;
```

Inny sposób wprowadzania atrybutu unikalności:

```
create table T1(Id int, Name char(25) unique);     (tak było poprzednio)
create table T2(Id int, Name char(25), constraint T2_unq unique (Name));
create table T3(Id int, Name char(25), constraint T3_unq unique (Id,Name));
desc T0;
desc T1;
desc T2;
desc T3;
```

W tym wypadku wprowadzamy więzy (constraint) o nadanej przez nas nazwie (tu T2_unq), typ więzów (unique) oraz kolumny, których więzy dotyczą. W ten sposób możemy zażądać unikalności w kilku kolumnach.

Wykonanie ostatniego polecenia nie informuje dokładnie o charakterze więzów. To możemy sprawdzić wykonując:

```
show index from T3;

albo

select column_name,constraint_name from information_schema.key_column_usage
where table_name = 'T3';


insert into T3 values (1024,'Nowak');
insert into T3 values (1024,'Nowak');   blad!

insert into T3 values (1025,'Nowak');   OK!
```

```
insert into T3 values (1025,'Kowalski');   OK
```

Więzy unikalności można nałożyć na istniejącą tabelę z danymi stosując polecenie

```
alter table ... add constraint ...
```

o ile dane te nie są sprzeczne z żądaniem unikalności:

```
drop table T0,T1,T2,T3;
create table T0(Id int, Name char(25));
insert T0 values (1001,'Nowak'),(1002,'Nowak');
alter table T0 add constraint T0_unq unique (Name);
#=== ERROR ===
desc T0;
```

Primary key

Klucz główny pozwala na jednoznaczną identyfikację krotek i w MySQL odpowiada łącznemu użyciu atrybutów not **null** i **unique**.

Ustawienie klucza głównego

Metoda 1

```
create table T1(
      Id int primary key,
      Name char(15)
);
```

```
desc T1;
```

Metoda 2

```
create table T1(
      Id int,
      Name char(15),
      primary key (Id)
);
```

```
desc T1;
```

Powyższe metody nie sg do końca równoważne (na czym polega różnica?)

```
insert T1 () values ();
```

```
select * from T1;
```

```
insert T1 () values ();
```

```
#=== ERROR ===
```

Druga metoda pozwala na definiowanie klucza głównego opartego na więcej niż jednej kolumnie:

```
create table T2(
      Id int,
      Name char(15),
      primary key (Id,Name)

);

desc T2;
```

Klucz główny można dodać do istniejącej tabeli:

```
create table T3(
      Id int,
      Name char(15)
);

alter table T3 add primary key (Id);

desc T3;
```

Usunięcie klucza głównego

```
alter table T3 drop primary key;

desc T3;
```

Auto increment

Autoinkrementacja może być stosowana tylko w kolumnie klucza głównego i zapewnia automatyczną inkrementację jego wartości:

```
create table T3(
     Id int auto_increment,
     Name char(15),
     primary key (id)
);

desc T3;

insert into T3 () values ();
insert into T3 () values ();
insert into T3 () values ();

select * from T3;
```