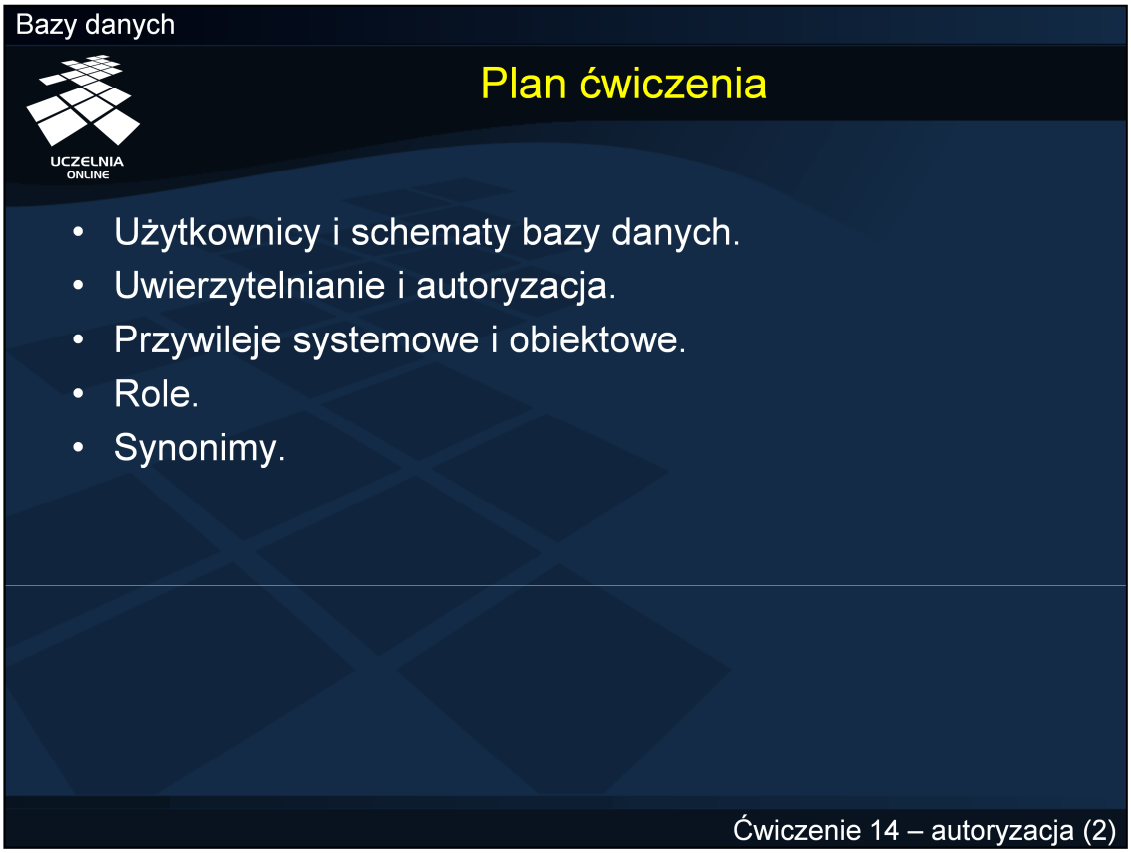
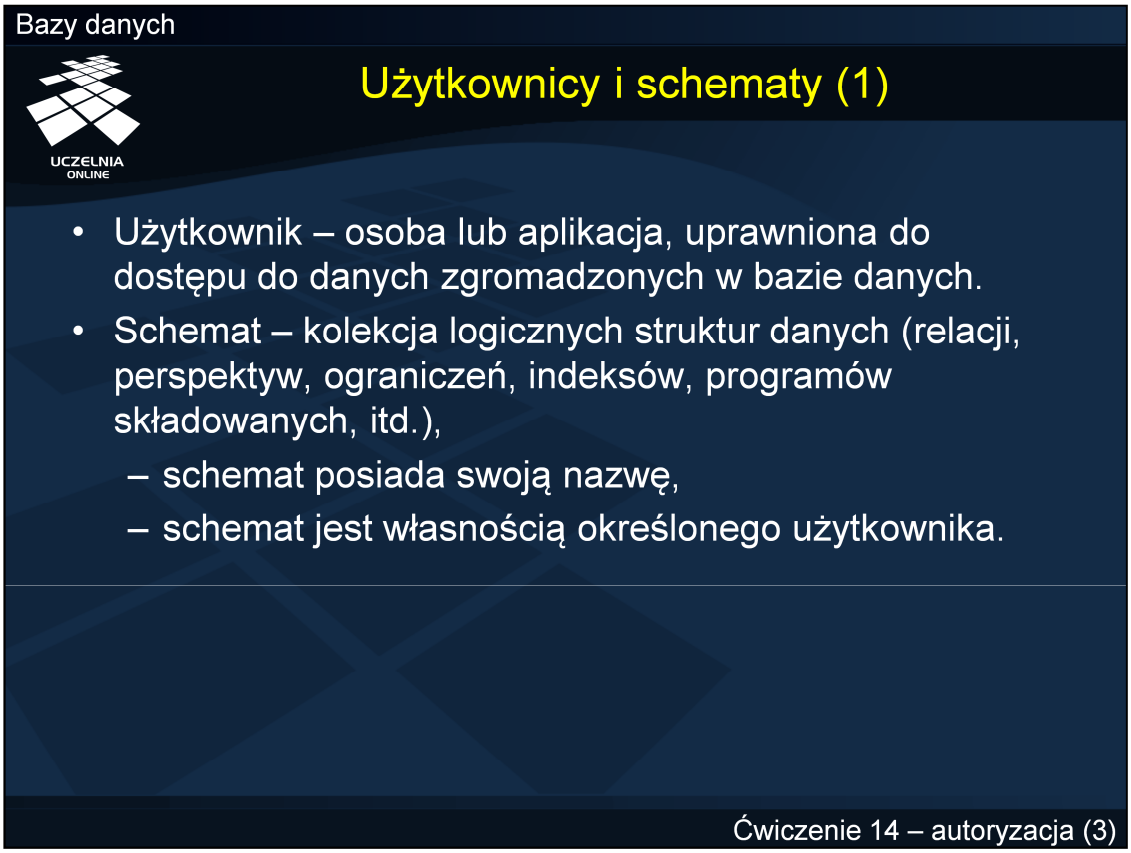
Niniejsze ćwiczenie zaprezentuje zagadnienia związane z systemem bezpieczeństwa bazy danych.

Wymagania:

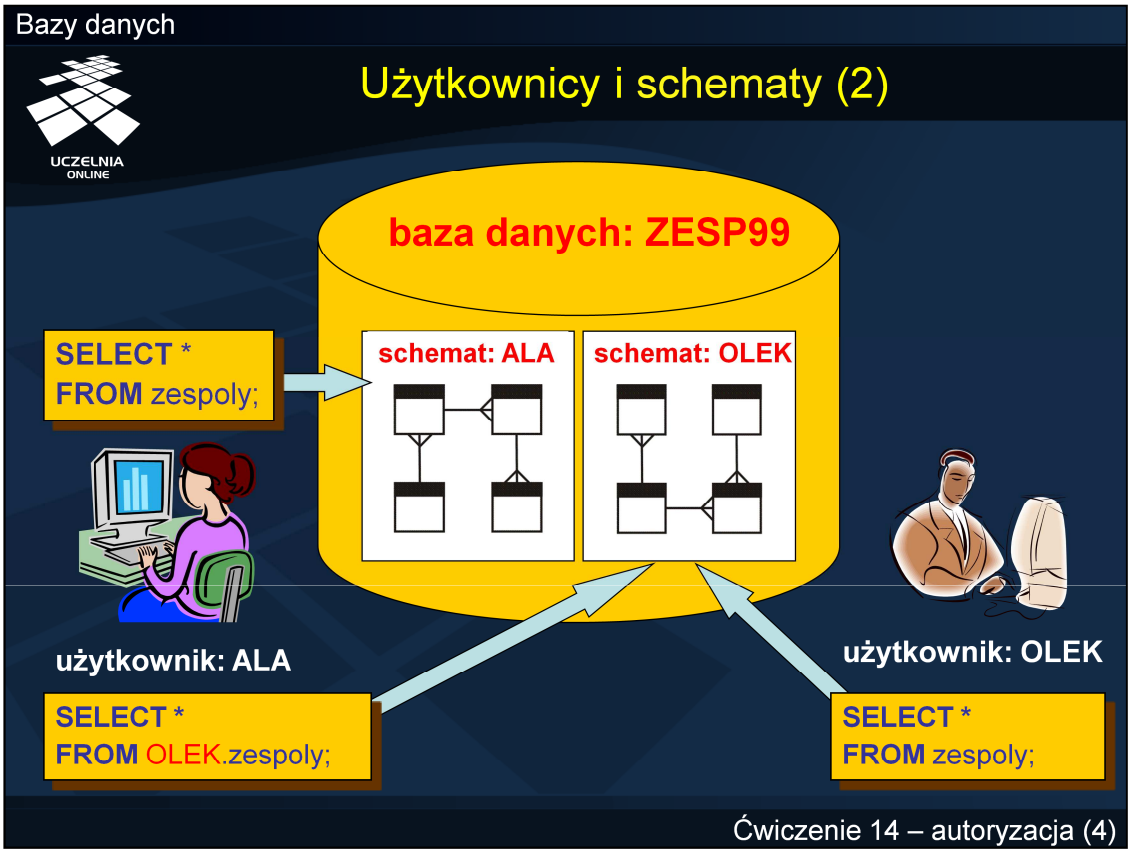
Umiejętność pisania prostych zapytań w języku SQL.



Na początku ćwiczenia zdefiniowane zostaną pojęcia „użytkownik” i „schemat” bazy danych. Następnie zaprezentowane będą mechanizmy uwierzytelniania użytkowników bazy danych. Dalej zajmiemy się koncepcją autoryzacji użytkowników w bazie danych i mechanizmami wykorzystywanymi w tym zakresie: przywilejami i rolami. Na końcu ćwiczenia omówione zostaną synonimy.

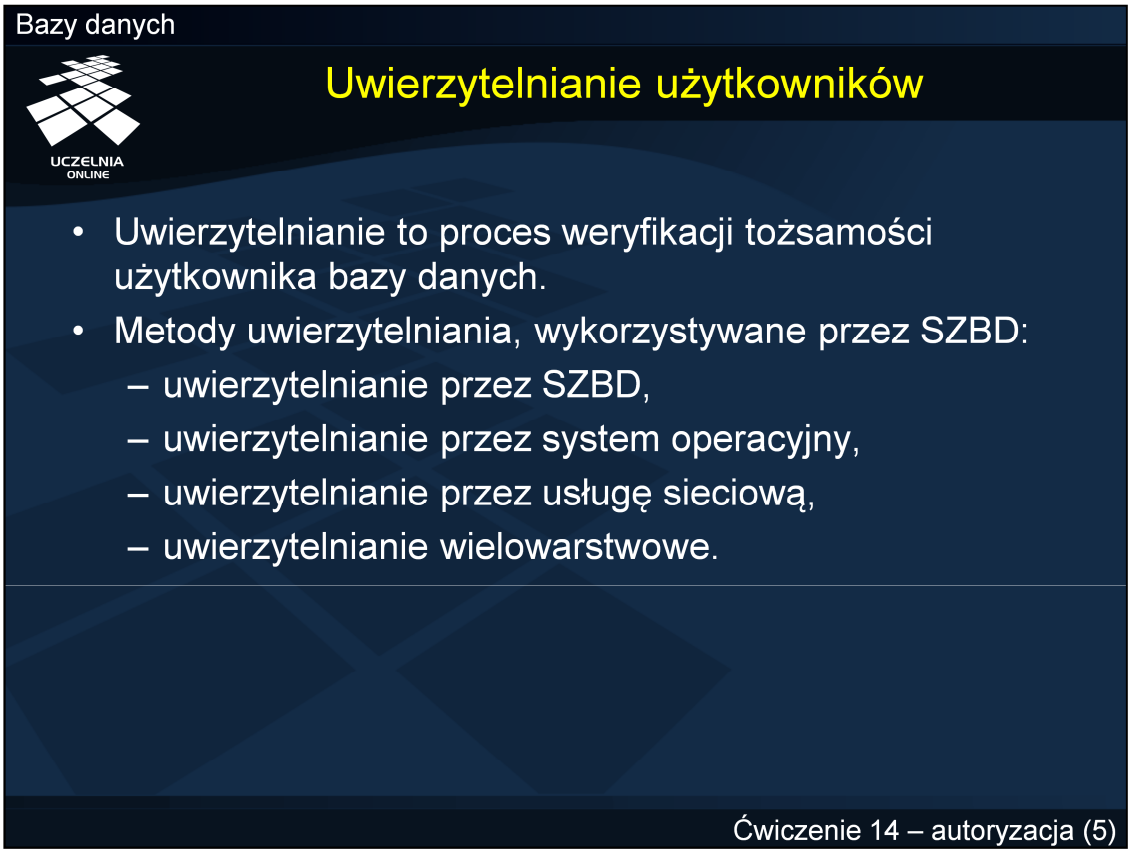


Pod pojęciem „użytkownik bazy danych” rozumiemy osobę lub aplikację, która jest uprawniona do dostępu do danych zgromadzonych w bazie danych. Każdy użytkownik ma przypisany schemat, który należy rozumieć jako konto użytkownika w bazie danych. Schemat jest kolekcją logicznych struktur danych, takich jak relacje, perspektywy, ograniczenia integralnościowe, indeksy, programy składowane i inne. Schemat posiada swoją własną nazwę, w niektórych systemach zarządzania bazami danych, np. w Oracle, nazwa użytkownika, będącego właścicielem schematu, jest identyczna z nazwą schematu.

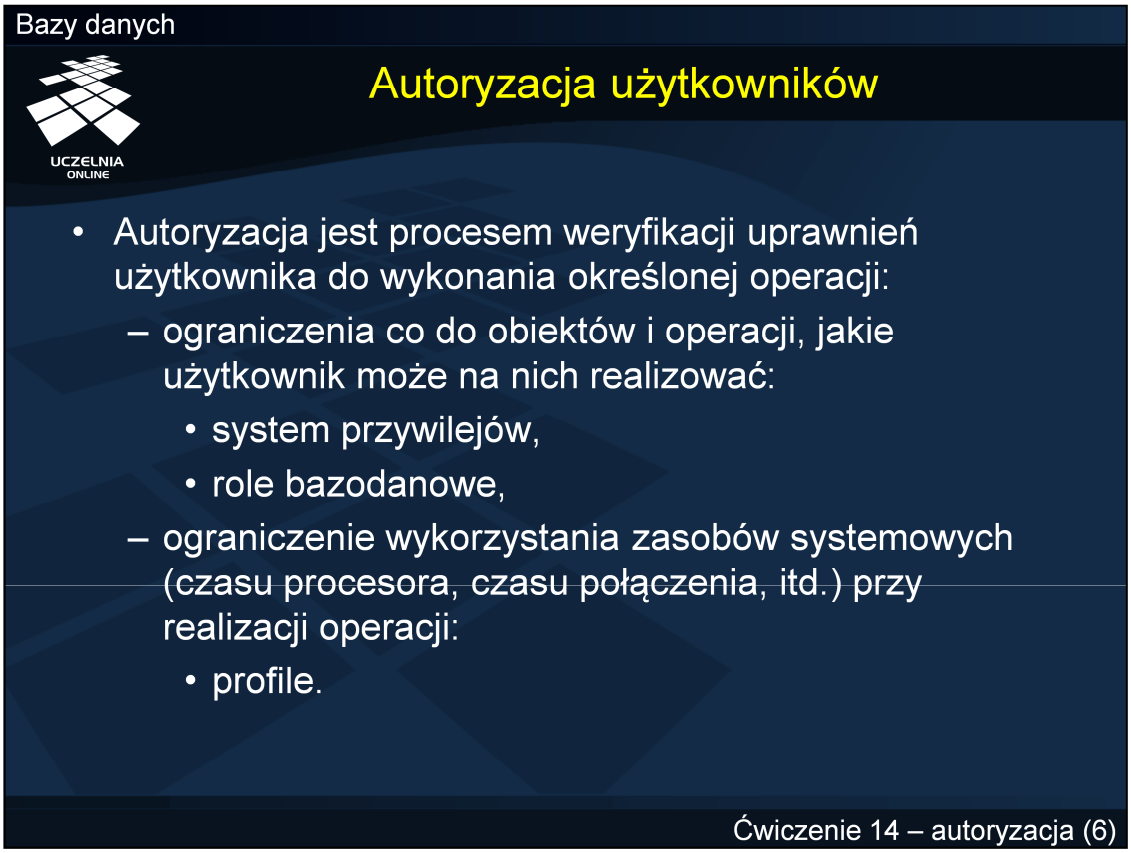


Na rysunku przedstawiono bazę danych o nazwie „ZESP99”. Baza składa się z dwóch schematów o nazwach „ALA” i „OLEK”, załóżmy, że w każdym ze schematów utworzono relacje, które używamy do ćwiczeń (relacje PRACOWNICY, ZESPOLY i ETATY). Właścicielami schematów „ALA” i „OLEK” jest dwoje użytkowników, odpowiednio ALA i OLEK. Domyślnie każdy z użytkowników ma możliwość wykonywania operacji na danych, przechowywanych w relacjach schematu, którego jest właścicielem. I tak użytkownik ALA wykonuje zapytanie do relacji ZESPOLY w schemacie „ALA”, wskazując w klauzuli FROM zapytania wprost relację ZESPOLY. Podobnie użytkownik OLEK wykonuje zapytanie do relacji ZESPOLY, utworzonej w schemacie „OLEK”, którego jest właścicielem. Załóżmy teraz, że użytkownik ALA chciałaby odczytać dane relacji ZESPOLY ze schematu użytkownika OLEK. W takim przypadku w klauzuli FROM zapytania musi umieścić przed nazwą relacji nazwę schematu, a więc posłużyć się konstrukcją „OLEK.ZESPOLY”. Aby odczyt się powiódł, użytkownik OLEK musi pozwolić użytkownikowi ALA na dostęp do relacji z jego schematu.

Reasumując, użytkownicy, wykonując dostęp do obiektów z własnych schematów, podają w poleceniach SQL nazwy obiektów bezpośrednio (mogą umieścić przed nazwą obiektu nazwę swojego schematu, jednak nie jest to konieczne). Jeśli jednak użytkownik chce wykonać dostęp do obiektów, umieszczonych w schemacie innym niż jego własny, musi poprzedzić nazwę obiektu nazwą schematu. Dostęp zakończy się sukcesem, jeśli właściciel schematu pozwoli użytkownikowi na dostęp do obiektów, których jest właścicielem.

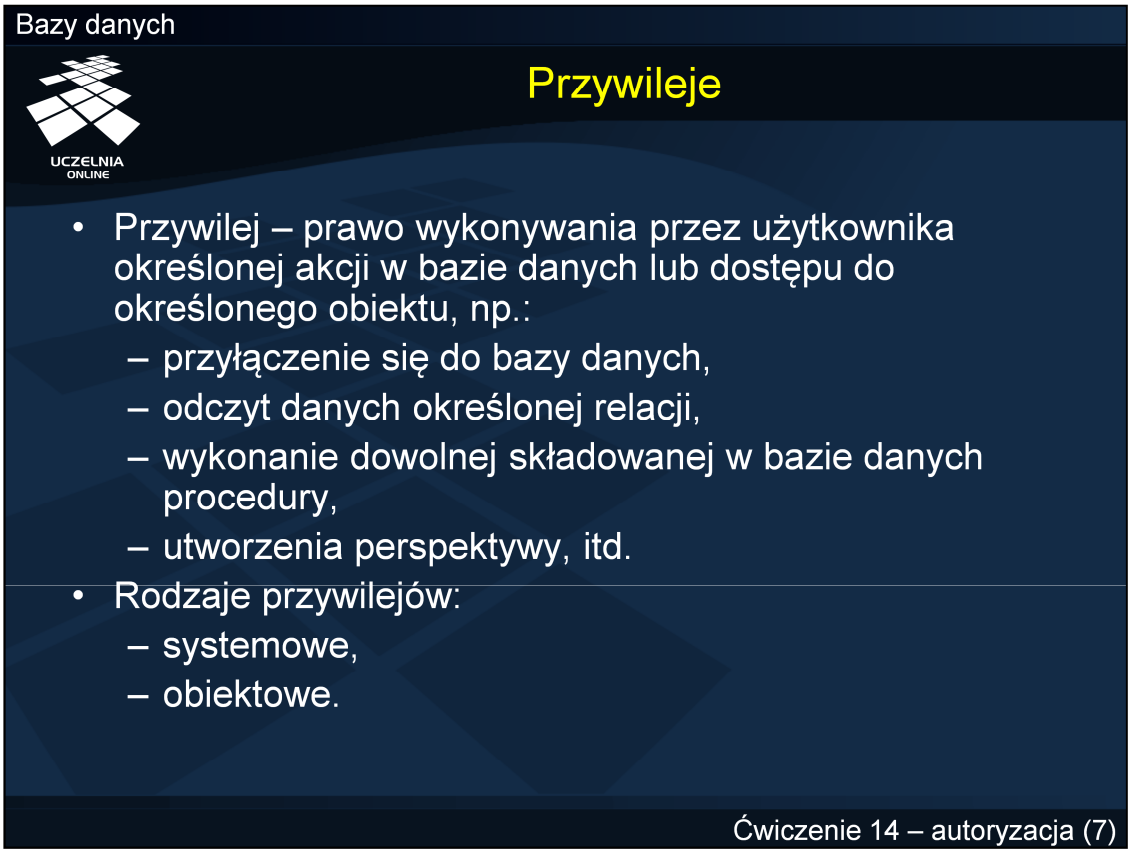


Omówimy teraz metody uwierzytelniania użytkowników bazy danych. Proces uwierzytelniania polega na weryfikacji tożsamości użytkownika, który chce uzyskać dostęp do bazy danych. Uwierzytelnianie może być realizowane kilkoma metodami. Pierwsza z nich to uwierzytelnianie przez bazę danych. Użytkownik, dołączając się do bazy danych, musi podać swój identyfikator i hasło. Wszystkie informacje, konieczne do weryfikacji tożsamości użytkownika (identyfikatory i hasła użytkowników), są składowane bezpośrednio w bazie danych. Kolejna metoda uwierzytelniania, uwierzytelnianie przez system operacyjny, wykorzystuje mechanizmy systemu operacyjnego do weryfikacji tożsamości użytkownika. Użytkownik dołącza się do bazy danych bez podania swojego identyfikatora i hasła, baza danych wykorzystuje informacje, które użytkownik podał, uwierzytelniając się przy rozpoczynaniu pracy z systemem operacyjnym. Uwierzytelnienie przez usługę sieciową polega na wykorzystaniu przez bazę danych funkcjonalności zewnętrznego oprogramowania, które bierze na siebie weryfikację użytkownika, chcącego przyłączyć się do bazy danych. Przykłady takich rozwiązań to Kerberos, RADIUS, wykorzystanie mechanizmu LDAP. W przypadku środowiska wielowarstwowego (np. z serwerem aplikacji) uwierzytelnianie może być realizowane przez inną warstwę środowiska niż baza danych, np. warstwę pośrednią, serwer aplikacji. Rozwiązanie to może mieć kilka dodatkowych zalet, np. zastosowanie puli połączeń (ang. connection pooling), gdzie wielu użytkowników korzysta ze współdzielonych połączeń do bazy danych zamiast połączeń dedykowanych.



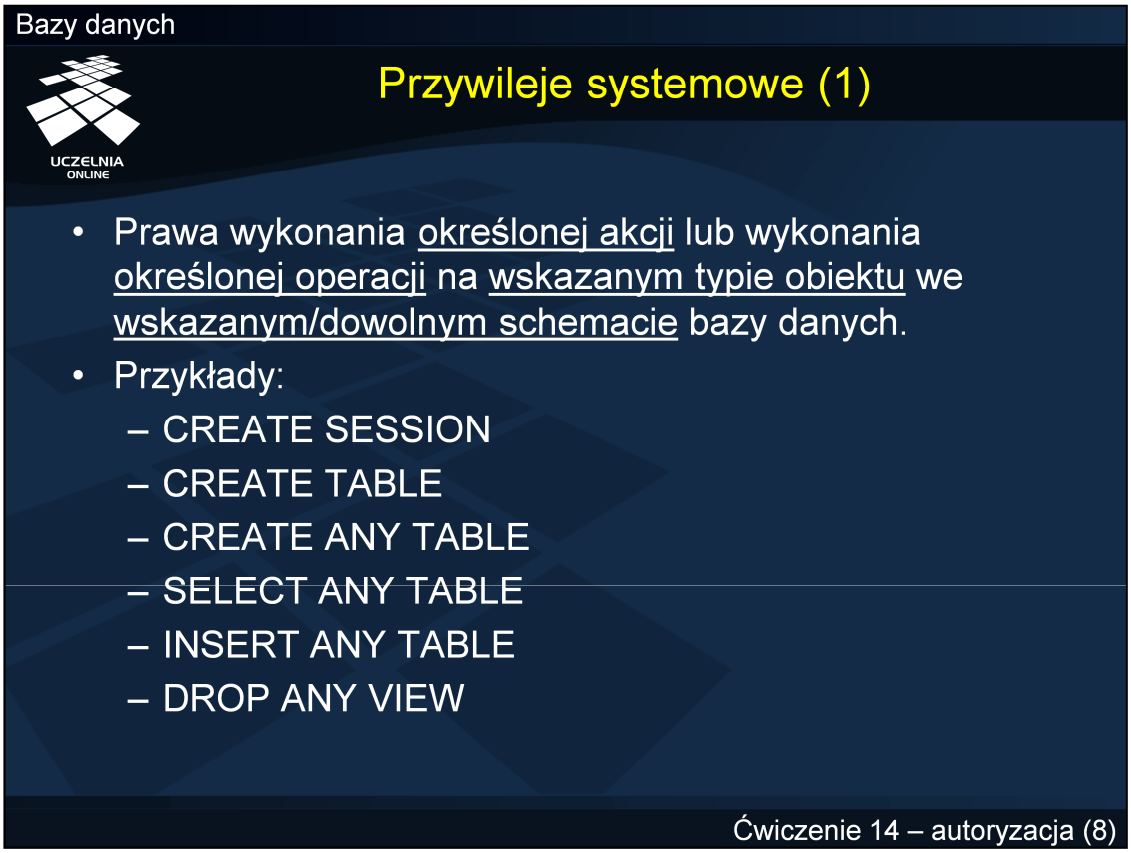
Użytkownik, który pomyślnie przeszedł proces uwierzytelniania, uzyskuje możliwość przyłączenia się do bazy danych. Zakres działań, które użytkownik może realizować w bazie danych, jest przedmiotem kolejnego procesu, tzw. autoryzacji. Autoryzacja użytkownika polega na weryfikacji uprawnień użytkownika do realizacji określonych operacji w bazie danych. Działania użytkownika mogą być ograniczane w dwóch zakresach. Pierwszy z nich to możliwość wykonywania przez użytkownika tylko określonego zbioru operacji w bazie danych, dodatkowo na określonym zbiorze obiektów. Mechanizmami, wykorzystywanymi w tym celu, są: system przywilejów bazy danych i system ról bazy danych. Zagadnienia te będą stanowiły przedmiot dalszych rozważań niniejszego ćwiczenia.

Drugi zakres może kontrolować wykorzystanie przez działania użytkownika zasobów systemowych bazy danych. Tutaj możliwe ograniczenia to czas procesora potrzebny do realizacji operacji użytkownika, ilość danych, odczytanych i zapisanych przez polecenia użytkownika, czy też czas trwania sesji użytkownika. Jednym z mechanizmów, umożliwiających kontrolę wykorzystania zasobów, są tzw. profile. Nie będą one jednak poruszane w niniejszym ćwiczeniu – zainteresowanych odsyłamy do dokumentacji technicznej SZBD Oracle.



Bieżący slajd rozpoczyna omawianie systemu przywilejów bazy danych. Przywilej jest prawem wykonania określonej akcji w bazie danych lub możliwością dostępu do określonego obiektu w bazie danych. Przykładami przywilejów są: możliwość przyłączenia się użytkownika do bazy danych, prawo odczytu danych określonej relacji, prawo wykonania dowolnej procedury składowanej w bazie danych (również tych spoza schematu użytkownika) czy też prawo utworzenia perspektywy. Tak więc każdy użytkownik posiada przydzielony mu zbiór przywilejów, które ograniczają zakres operacji możliwych do realizacji przez użytkownika w bazie danych.

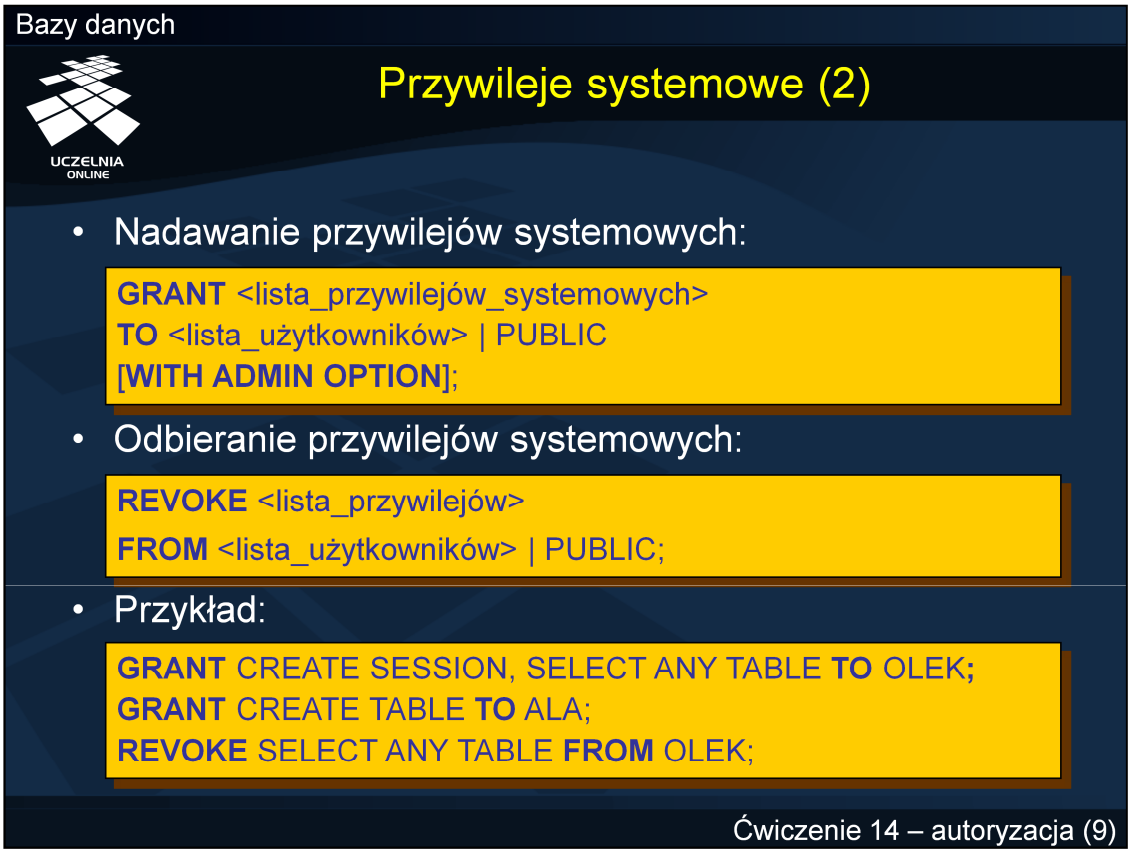
Przywileje dzielimy na dwa rodzaje: przywileje systemowe i przywileje obiektowe. Na następnym slajdzie omówimy przywileje systemowe.



Przywilej systemowy jest prawem wykonywania określonej akcji w bazie danych lub wykonania określonej operacji na wskazanym typie obiektu we wskazanym schemacie lub całej bazie danych. Przywilej systemowy nigdy nie specyfikuje konkretnego obiektu.

Lista dostępnych w bazie danych przywilejów systemowych jest różna i zależy od systemu zarządzania bazą danych. Np. w SZBD Oracle zbiór przywilejów systemowych liczy ponad 100 pozycji. Kilka z nich wymieniono na slajdzie. Przywilej CREATE SESSION umożliwia użytkownikowi przyłączenie się do bazy danych, lub mówiąc ściśle, utworzenie sesji w bazie danych. Bez tego przywileju użytkownik, mimo że został pomyślnie uwierzytelniony, nie zostanie przyłączony do bazy danych. Przywilej CREATE TABLE pozwala użytkownikowi na tworzenie relacji w schemacie, którego jest właścicielem. Z kolei przywilej CREATE ANY TABLE daje użytkownikowi prawo tworzenia relacji w dowolnym schemacie bazy danych. Dwa kolejne przywileje, SELECT ANY TABLE i INSERT ANY TABLE umożliwiają użytkownikowi, odpowiednio, wydawanie zapytań i dodawanie rekordów do dowolnej relacji w bazie danych. Ostatni z zaprezentowanych przywilejów, DROP ANY VIEW, umożliwia użytkownikowi usunięcie definicji dowolnej perspektywy z bazy danych.

Jak widzimy z zaprezentowanych przykładów, przywileje systemowe dają użytkownikom możliwość realizacji szerokiego zakresu operacji w bazie danych, dlatego należy je przydzielać ostrożnie.

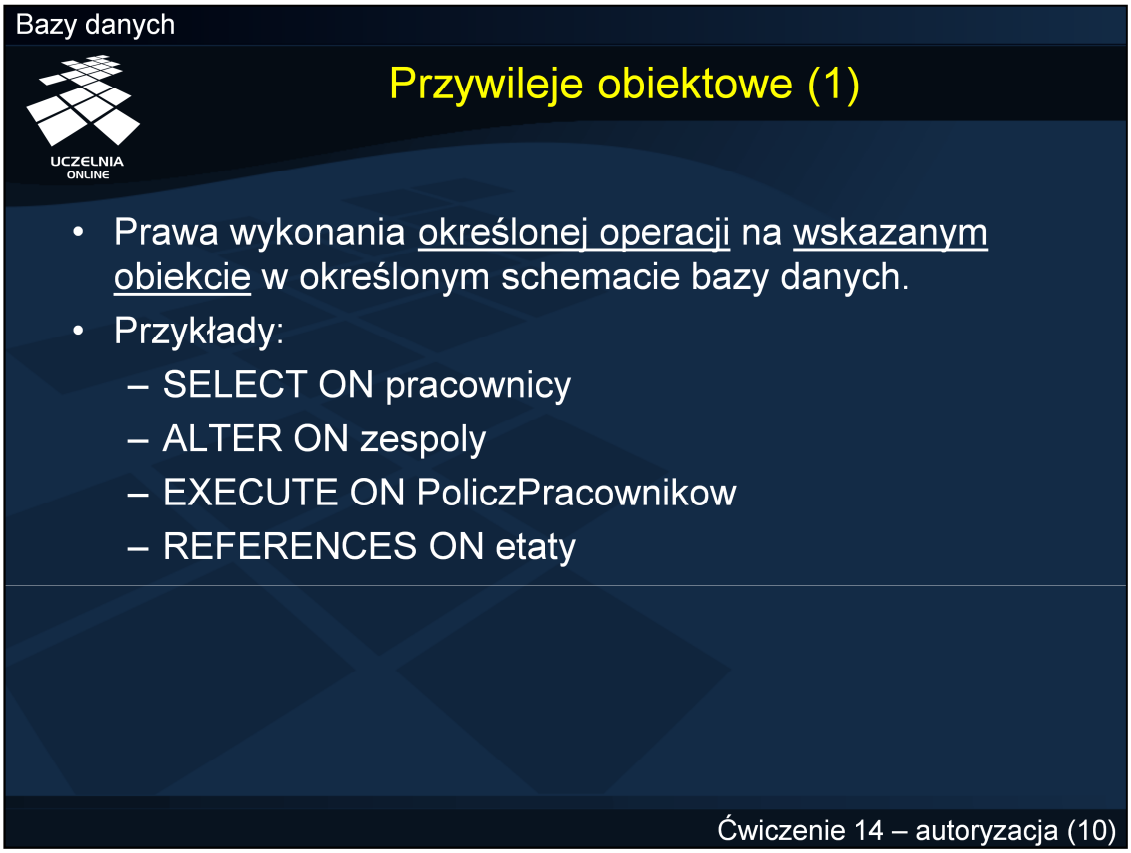


Bieżący slajd pokazuje zbiór poleceń umożliwiających nadawania i odbieranie przywilejów systemowych.

Polecenie GRANT służy do nadawania użytkownikom przywilejów systemowych. Po słowie GRANT podaje się listę przywilejów systemowych, oddzielonych przecinkami, następnie po słowie TO podaje się listę nazw użytkowników, również oddzielonych przecinkami, którzy mają wymienione przywileje systemowe otrzymać. Jeśli przywileje mają zostać przydzielone wszystkim użytkownikom bazy danych, można posłużyć się słowem PUBLIC, określającym grupę, do której należą wszyscy użytkownicy bazy danych. Dodatkowa klauzula WITH ADMIN OPTION umożliwia użytkownikowi, który przywilej otrzymał, przekazanie go innym użytkownikom (przez wydanie kolejnego polecenia GRANT).

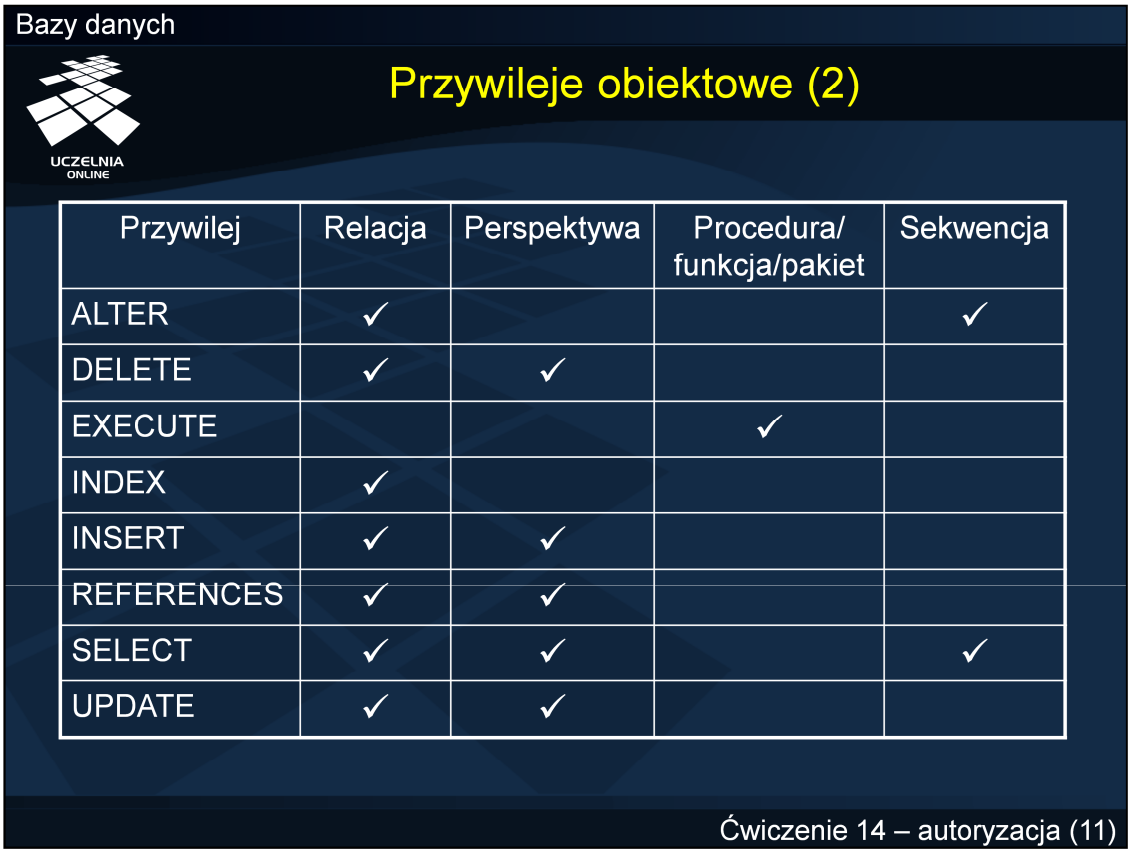
Wykonując polecenie REVOKE możemy odebrać przydzielone wcześniej użytkownikom przywileje systemowe. Listę odbieranych przywilejów umieszczamy po słowie REVOKE, po słowie FROM podajemy listę nazw użytkowników, którym wymienione przywileje zostaną odebrane. Jeśli chcemy odebrać określone przywileje wszystkim użytkownikom, możemy zamiast listy użytkowników wskazać grupę PUBLIC.

Omówmy zaprezentowane na slajdzie przykłady. Pierwsze polecenie nadaje użytkownikowi OLEK prawo przyłączenia się do bazy danych i możliwość odczytu danych dowolnej relacji z bazy danych. Drugie polecenie nadaje użytkownikowi ALA prawo tworzenia relacji w jej własnym schemacie. Ostatnie polecenie odbiera użytkownikowi OLEK przydzielone wcześniej prawo odczytu danych dowolnej relacji z bazy danych.



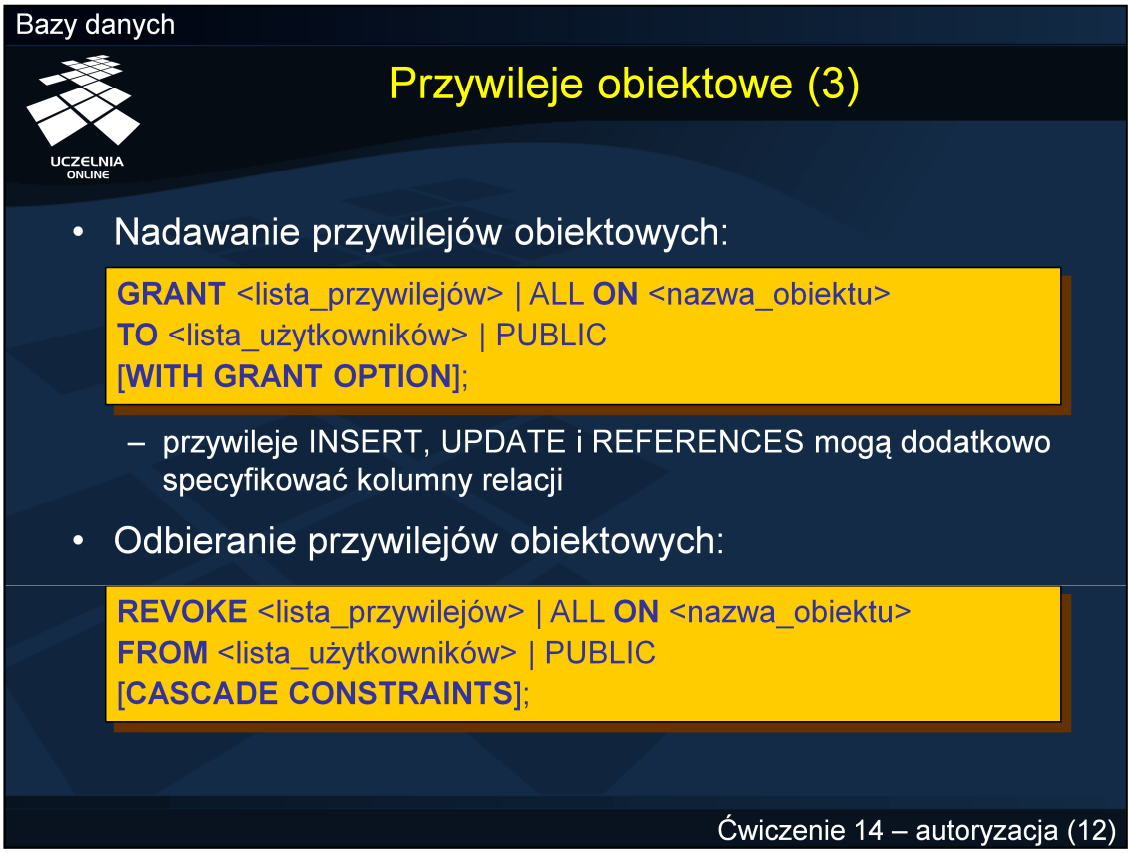
Drugi rodzaj przywilejów, przywileje obiektowe, określają uprawnienia użytkownika do wykonania określonej operacji na wskazanym obiekcie, znajdującym się w określonym schemacie bazy danych.

Przykłady przywilejów to: SELECT ON PRACOWNICY pozwala użytkownikowi, który otrzymał ten przywilej, na odczyt danych relacji PRACOWNICY ze schematu użytkownika, który ten przywilej nadał, ALTER ON ZESPOLY umożliwia temu użytkownikowi modyfikację struktury relacji ZESPOLY, znajdującej się w schemacie użytkownika, nadającego przywilej, EXECUTE ON PoliczPracownikow daje użytkownikowi możliwość wykonania programu składowanego o podanej nazwie, umieszczonego w schemacie użytkownika nadającego ten przywilej, z kolei REFERENCES ON ETATY daje użytkownikowi możliwość przyłączenia się do relacji ETATY kluczem obcym.



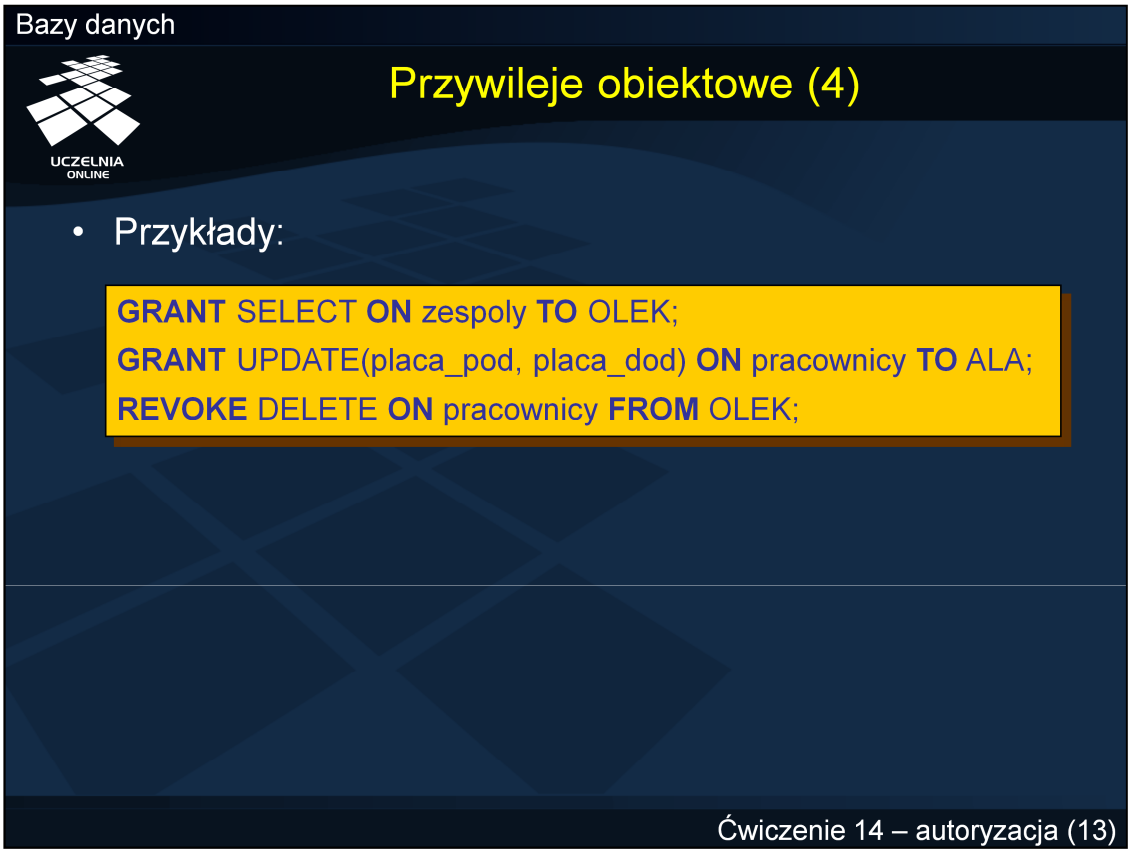
Przywilej	Relacja	Perspektywa	Procedura/ funkcja/pakiet	Sekwencja
ALTER	✓			✓
DELETE	✓	✓		
EXECUTE			✓	
INDEX	✓			
INSERT	✓	✓		
REFERENCES	✓	✓		
SELECT	✓	✓		✓
UPDATE	✓	✓		

Przywileje obiektowe stosuje się do określonych obiektów bazodanowych. Przywilej ALTER pozwala na modyfikację struktur określonych relacji lub sekwencji. Przywilej DELETE daje użytkownikowi możliwość usuwania danych ze wskazanej relacji lub perspektywy. Przywilej EXECUTE stosuje się tylko do kodu składowanego – umożliwia wywołanie określonej funkcji, procedury lub pakietu. Przywilej INDEX stosuje się do relacji celem wskazania, że użytkownik może zakładać na tej relacji indeksy. Przywilej INSERT pozwala na wstawianie rekordów do określonej relacji lub perspektywy. Przywilej REFERENCES daje użytkownikowi możliwość wykorzystania określonej relacji lub perspektywy do przyłączenia kluczem obcym z innej relacji lub perspektywy. Przywilej SELECT umożliwia wydawanie zapytań do wskazanej relacji lub perspektyw i odczyt wartości wskazanej sekwencji. Przywilej UPDATE daje możliwość modyfikacji danych określonej relacji lub perspektywy.

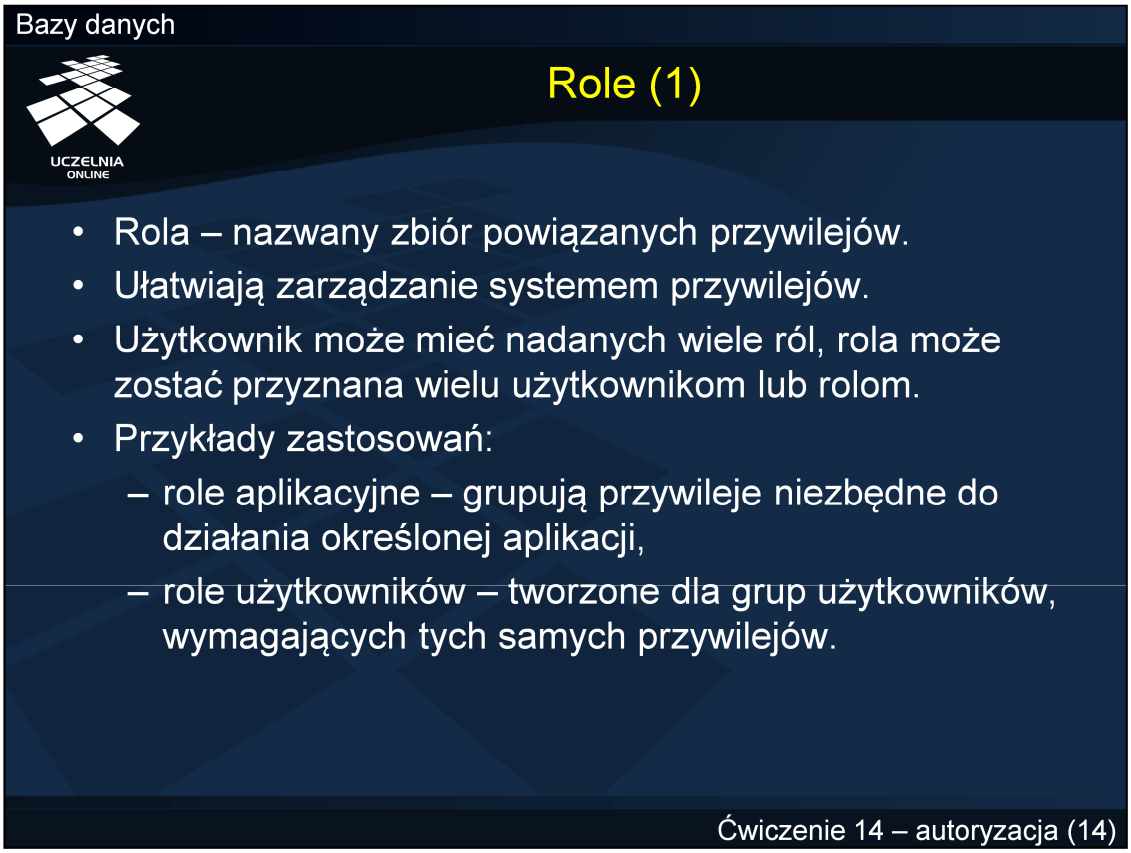


Do nadawania przywilejów obiektowych służy, podobnie jak to było w przypadku przywilejów systemowych, polecenie GRANT. Po słowie GRANT podajemy listę przywilejów obiektowych, które mają zostać przyznane, następnie, po słowie ON, podajemy nazwę obiektu, którego przywileje dotyczą. Dla przywilejów INSERT, UPDATE i REFERENCES można dodatkowo ograniczyć przywilej do listy kolumn (np. możliwość modyfikowania wartości jedynie kolumn PLACA_POD i PLACAD_DOD relacji PRACOWNICY). Jeśli chcemy nadać wszystkie przywileje charakterystyczne dla danego obiektu, możemy użyć słowa ALL (np. dla relacji ALL będzie oznaczać przywileje: ALTER, DELETE, INDEX, INSERT, REFERENCES, SELECT, UPDATE, natomiast ALL dla procedury to tylko EXECUTE). Po słowie TO umieszczamy listę użytkowników, którym przyznane zostaną przywileje, ewentualnie słowo PUBLIC, jeśli przywileje mają być przyznane wszystkim użytkownikom bazy danych. Opcjonalna klauzula WITH GRANT OPTION umożliwia nadanie użytkownikom prawa przekazywania otrzymanych przywilejów innym użytkownikom (przez wykonanie kolejnego polecenia GRANT).

Odebranie użytkownikom nadanych wcześniej przywilejów obiektowych realizuje się poleceniem REVOKE. Po słowie REVOKE podajemy listę odbieranych przywilejów lub słowo ALL jeśli chcemy odebrać wszystkie przyznane przywileje, po słowie ON podajemy nazwę obiektu, którego przywileje dotyczą, a po słowie FROM listę użytkowników, którym przywileje odbieramy lub grupę PUBLIC. Dla przywileju REFERENCES można podać dodatkową klauzulę CASCADE CONSTRAINTS, która spowoduje automatyczne usunięcie kluczy obcych, założonych przez użytkownika w czasie posiadania przywileju.

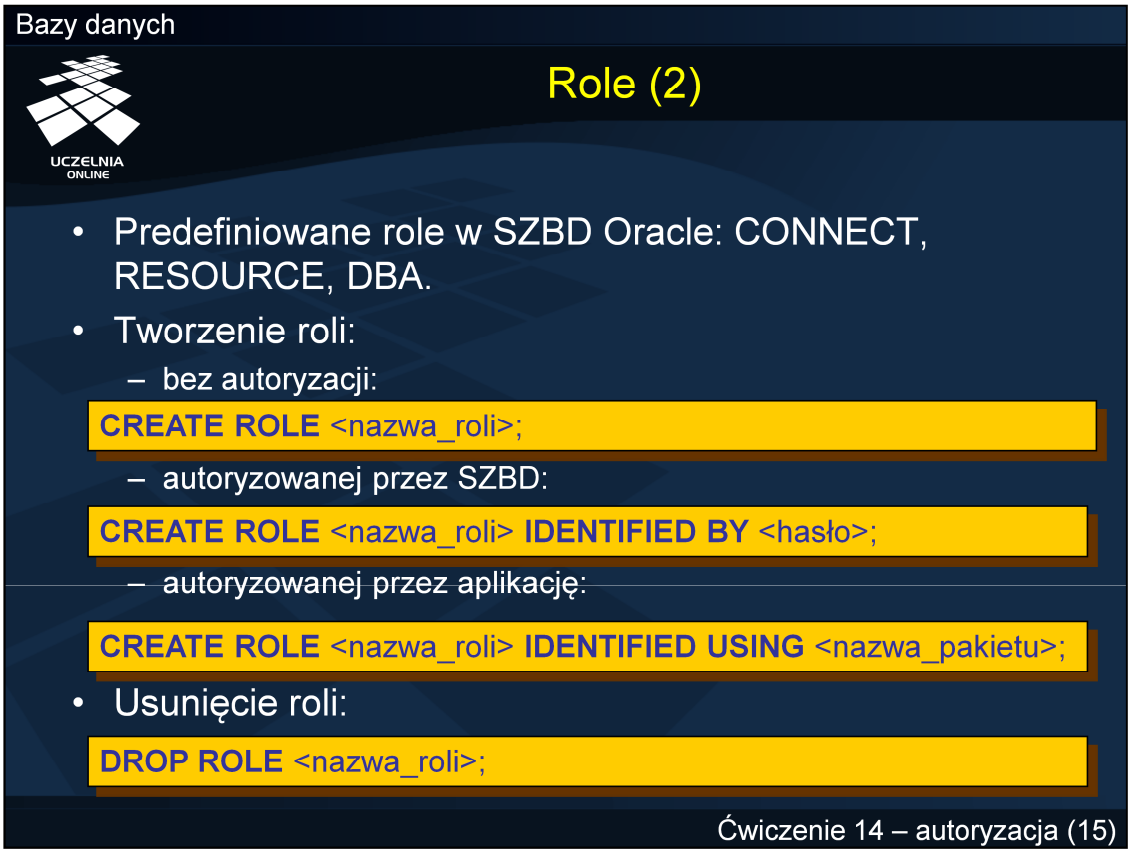


Bieżący slajd przedstawia sekwencję przykładowych poleceń. Pierwsze z nich wykonał użytkownik, będący właścicielem relacji ZESPOLY, nadając użytkownikowi OLEK prawo wykonywania zapytań do tej relacji. W drugim poleceniu właściciel relacji PRACOWNICY nadał użytkownikowi ALA prawo modyfikacji danych relacji PRACOWNICY, z ograniczeniem do atrybutów PLACA_POD i PLACA_DOD. W ostatnim poleceniu właściciel relacji PRACOWNICY odebrał użytkownikowi OLEK prawo usuwania rekordów z tej relacji.



Rola jest nazwanym zbiorem przywilejów, zarówno systemowych, jak i obiektowych. Role znacznie ułatwiają zarządzanie przywilejami w bazie danych. Wyobraźmy sobie system informatyczny dużego przedsiębiorstwa, z którego korzysta kilkaset osób. Poszczególne osoby mają różne uprawnienia w dostępie do danych z bazy danych systemu. Administrator musi po kolei nadać każdemu użytkownikowi zbiór odpowiednich przywilejów. Znacznie prościej jest jednak połączyć użytkowników w grupy ze względu na posiadane przywileje (np. dyrektorzy, kierownicy działów, kierownicy sekcji, szeregowi pracownicy, itd.). Dla każdej grupy administrator tworzy rolę (np. „KIEROWNIK_DZIAŁU” dla grupy kierownicy działów), następnie do każdej z grup przydziela przywileje, które powinni otrzymać pracownicy danej grupy. Na końcu administrator przydziela poszczególnym użytkownikom odpowiednie role (np. użytkownik Kowalski, który jest kierownikiem działu, otrzymuje rolę „KIEROWNIK_DZIAŁU”). W konsekwencji użytkownicy otrzymują prawa przydzielone wcześniej do ról, które im przyznano. Każda zmiana zestawu przywilejów roli jest automatycznie propagowana do użytkowników, którzy daną rolę otrzymali. Użytkownik może mieć przydzielone jednocześnie wiele ról. Role mogą tworzyć hierarchię – danej roli można oprócz uprawnień przydzielić inną rolę (np. rola „KIEROWNIK_SEKCJI” ma przypisaną rolę „SZEREGOWY_PRACOWNIK” plus dodatkowe przywileje). Opisany powyżej przykład to tzw. role użytkowników.

Inne zastosowanie ról to tzw. role aplikacyjne. Role aplikacyjne grupują przywileje, niezbędne do działania w bazie danych określonej aplikacji. Role aplikacyjne nadaje się innym rolom lub konkretnym użytkownikom, którzy mają mieć prawo uruchamiania danej aplikacji. Dla określonej aplikacji możliwe jest utworzenie kilku różnych ról, określających różne zakresy danych, do których dostęp będzie możliwy przy użyciu aplikacji.



Systemy zarządzania bazami danych często posiadają zestaw predefiniowanych ról systemowych. Np. w SZBD Oracle rola CONNECT zawiera przywilej systemowy CREATE SESSION, rola RESOURCE zawiera przywileje systemowe pozwalające użytkownikowi tworzyć relacje, indeksy, sekwencje, procedury i funkcje składowane w swoim własnym schemacie, natomiast rola DBA jest zbiorem przywilejów dla użytkownika – administratora bazy danych.

Polecenie tworzenia roli ma różną postać w zależności od sposobu autoryzacji roli. Jeśli rola nie ma mieć autoryzacji (użytkownik, który otrzymuje rolę, od razu uzyskuje uprawnienia roli), używamy polecenia CREATE ROLE <nazwa_roli>. Z kolei rola autoryzowana przez system zarządzania bazą danych wymaga od użytkownika, który ją otrzymuje, wykonania polecenia włączającego rolę. Konieczne wtedy jest podanie hasła – dopiero wtedy użytkownik otrzymuje przywileje roli. Polecenie tworzące rolę autoryzowaną przez SZBD to CREATE ROLE <nazwa_roli> IDENTIFIED BY <hasło>. Ostatni rodzaj autoryzacji roli, autoryzacja przez aplikację, pozwala na odblokowanie roli jedynie przez aplikację, używającą autoryzowanego pakietu (pakiety zostały dokładnie omówione w ćwiczeniu 13.). W tym przypadku po nazwie roli po klauzuli IDENTIFIED USING podaje się nazwę odpowiedniego pakietu.

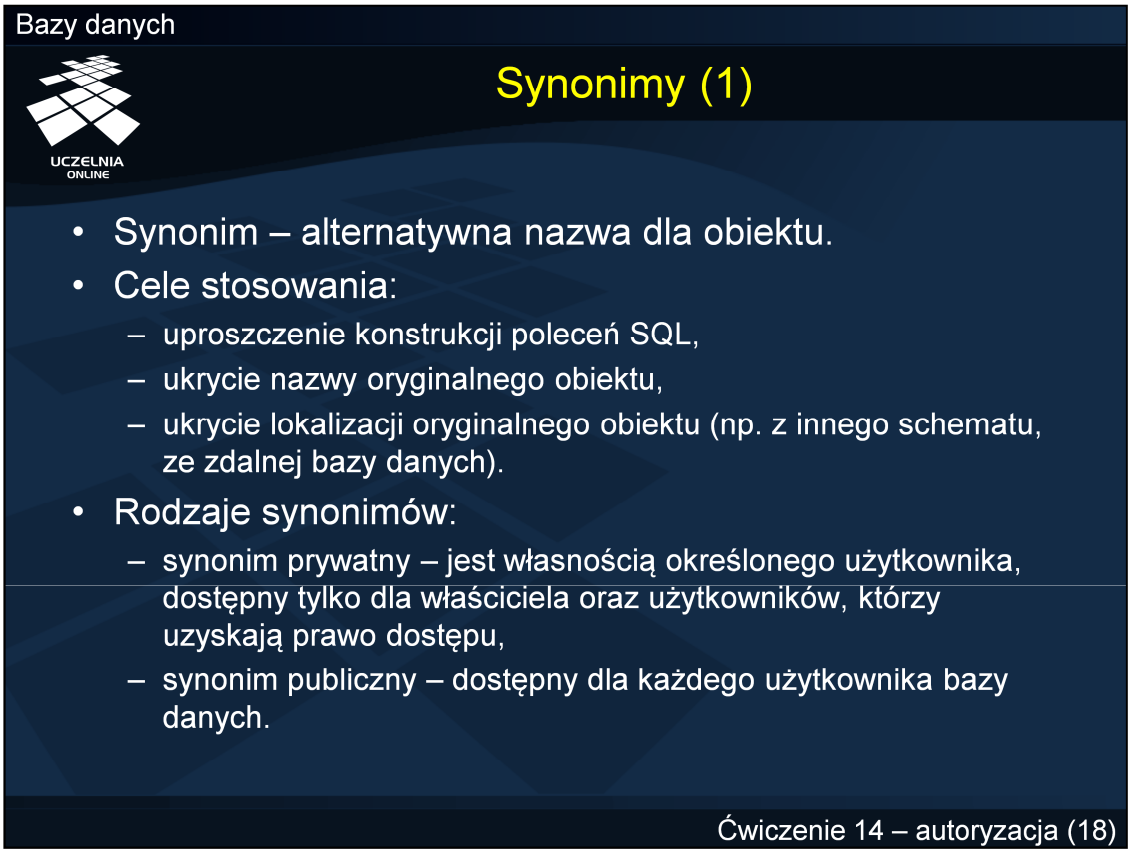
Usunięcie roli jest realizowane poleceniem DROP ROLE <nazwa_roli>.

Przyznanie roli zestawu przywilejów realizuje się takim samym poleceniem GRANT, jakie służy do przyznawania ról użytkownikom. Po słowie GRANT podajemy listę przywilejów, w przypadku przywilejów obiektowych konieczne jest podanie po słowie ON nazwy obiektu, którego przywileje dotyczą. Po słowie TO podajemy listę ról, które dane przywileje mają otrzymać. Polecenie GRANT służy również nadaniu roli użytkownikowi. Po słowie GRANT podajemy listę ról, po słowie TO listę użytkowników lub listę ról, które mają dane role otrzymać. Jeśli dana rola ma zostać przydzielona wszystkim użytkownikom, możemy użyć grupy PUBLIC. Jeśli użytkownicy mają mieć możliwość przekazywania otrzymanych ról innym użytkownikom bądź rolom, do polecenia dodaje się klauzulę WITH ADMIN OPTION.

Odbieranie przywilejów rolom i odbieranie ról użytkownikom bądź innym rolom to znane już nam polecenie REVOKE. W pierwszym przypadku po słowie REVOKE podajemy listę przywilejów, które mają zostać odebrane rolom, jeśli odbieramy przywileje obiektowe, po słowie ON podajemy nazwę obiektu. Nazwy ról umieszczamy po słowie FROM. W przypadku odbierania ról użytkownikom bądź rolom po słowie REVOKE umieszczamy listę ról, a po słowie FROM listę użytkowników bądź ról, którym określone wcześniej role mają zostać odebrane. Jeśli role mają zostać odebrane wszystkim użytkownikom bazy danych, możemy odwołać się do grupy PUBLIC.

Polecenie SET ROLE pozwala użytkownikowi w bieżącej sesji włączyć bądź wyłączyć daną rolę – w takiej sytuacji użytkownik otrzymuje zbiór przywilejów (włączenie roli) bądź przywileje traci (wyłączenie roli). Włączenie roli to polecenie SET ROLE <nazwa_roli>, jeśli rola jest autoryzowana przez bazę danych, konieczne jest jeszcze podanie hasła po klauzuli IDENTIFIED BY. Z kolei polecenie SET ROLE <nazwa_roli> ALL powoduje włączenie w bieżącej sesji wszystkich ról, przydzielonych wcześniej użytkownikowi, oprócz tych, które wymienione zostaną w opcjonalnej klauzuli EXCEPT. To polecenie nie może być jednak stosowane do ról autoryzowanych przez bazę danych. Z kolei polecenie SET ROLE NONE powoduje wyłączenie w bieżącej sesji wszystkich ról, które zostały przyznane użytkownikowi.

Zanalizujmy teraz sekwencję przykładowych operacji z bieżącego slajdu. Pierwsze polecenie, zakładamy, że wykonuje je właściciel relacji PRACOWNICY, tworzy rolę o nazwie KASJER. Jest to rola autoryzowana przez bazę danych, do jej aktywacji konieczne jest podanie hasła „pass123”. Następnie użytkownik przydziela roli KASJER prawa odczytu, modyfikacji i usuwania rekordów z relacji PRACOWNICY. W ostatnim poleceniu nadano rolę KASJER użytkownikowi o identyfikatorze KOWALSKI. Jednak użytkownik KOWALSKI nie otrzymuje od razu uprawnień roli. Musi najpierw w swojej sesji wykonać polecenie włączenia roli z podaniem hasła – dopiero wtedy może wykonywać operacje odczytu, modyfikacji i usuwania rekordów z relacji PRACOWNICY.

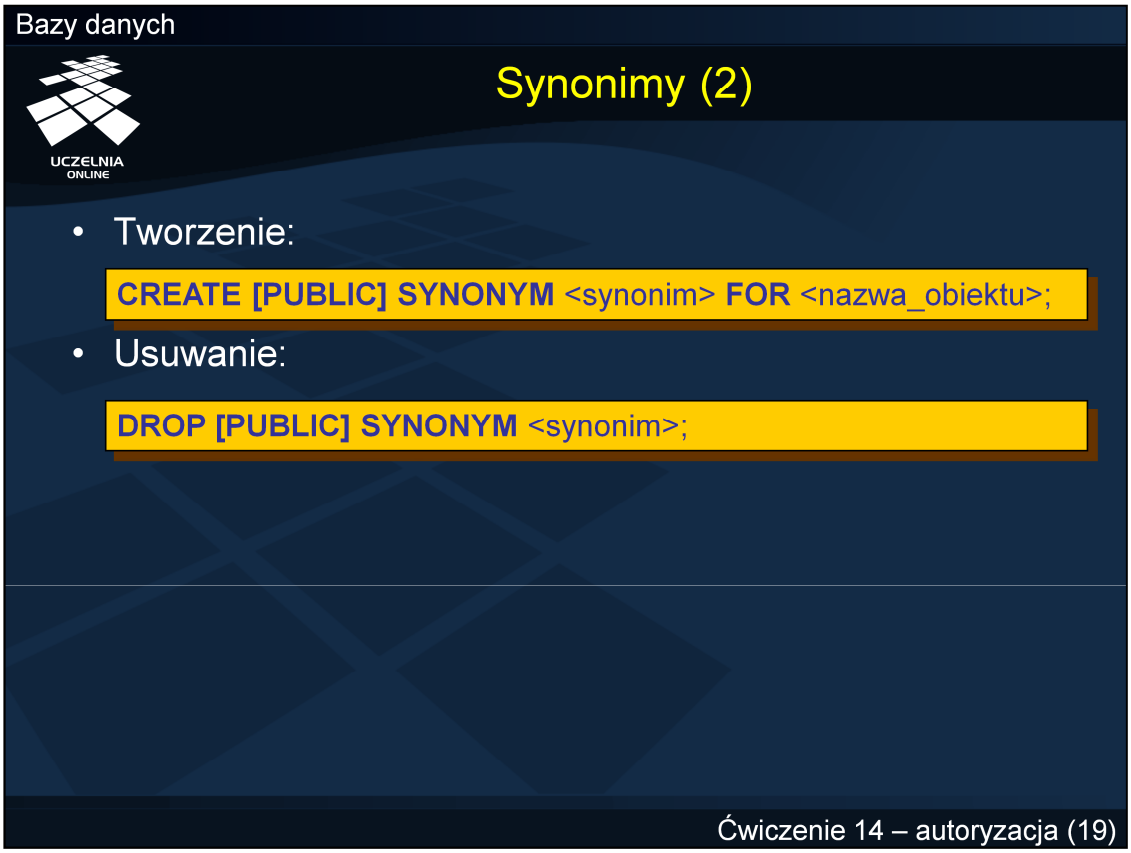


Omówimy teraz synonimy. Nie są to obiekty bezpośrednio związane z autoryzacją użytkowników bazy danych, są często wykorzystywane do znacznego uproszczenia postaci zapytań.

Synonim jest dodatkową nazwą dla obiektu, stosowaną w poleceniach SQL. Można podać wiele celów stosowania synonimów. Pierwszy z nich to uproszczenie konstrukcji poleceń SQL – zamiast używać w zapytaniu oryginalnej, niewygodnej nazwy relacji (np. PRAC1999-09-01) można dla niej utworzyć synonim (np. PRAC_WRZ). Synonim skutecznie ukrywa nazwę oryginalnego obiektu – użytkownik korzystający z synonimu nie ma często świadomości, że nazwa obiektu, z którego np. odczytuje dane, jest zupełnie inna. Wreszcie synonimy często stosuje się do uproszczenia dostępu do obiektów z innych schematów: zamiast w zapytaniach używać nazwy relacji z przedrostkiem w postaci nazwy schematu (np. OLEK.PRACOWNICY) łatwiej jest utworzyć synonim (np. PRAC).

Istnieją dwa rodzaje synonimów. Synonim prywatny może być utworzony przez użytkownika bazy danych i jest widoczny tylko dla swojego właściciela oraz dla użytkowników, którzy uzyskają prawo korzystania z synonimu. Stąd w bazie danych w różnych schematach może istnieć wiele synonimów prywatnych o tych samych nazwach. Z kolei synonim publiczny jest dostępny dla wszystkich użytkowników bazy danych, stąd jego nazwa musi być unikalna.

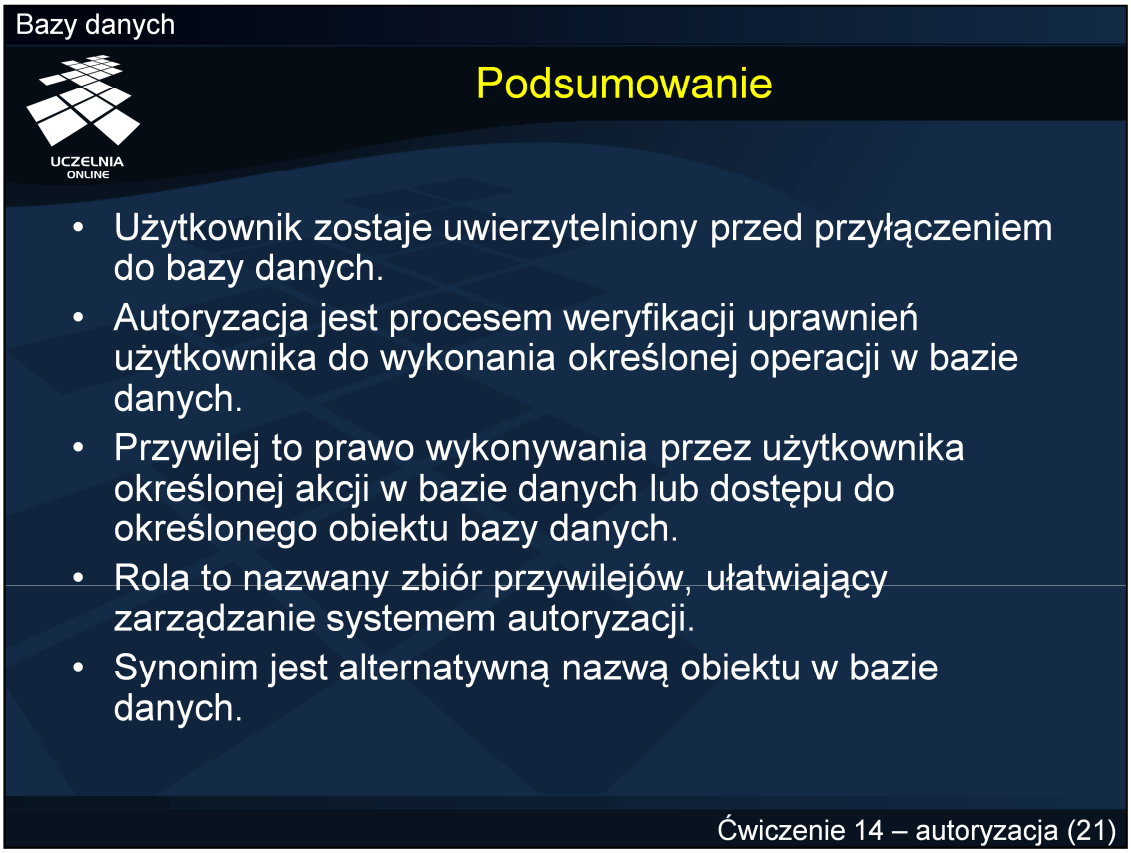
Należy podkreślić, że synonim nie jest w żaden sposób powiązany z obiektem, na który wskazuje. Gdy obiekt, dla którego synonim utworzono, zostaje usunięty, definicja synonimu pozostaje w bazie danych, ale synonim nie jest już dalej poprawny.



Bieżący slajd przedstawia polecenia zarządzania synonimami. Aby utworzyć synonim prywatny, po klauzuli CREATE SYNONYM podajemy nazwę synonimu, a po słowie FOR nazwę obiektu, na który synonim będzie wskazywał. Jeśli tworzony synonim ma być synonimem publicznym, po słowie CREATE należy dodać słowo PUBLIC.

Do usunięcia synonimu służy polecenie DROP SYNONYM, po słowie SYNONYM podajemy nazwę usuwanego synonimu. Przy usuwaniu synonimu publicznego konieczne jest dodanie słowa PUBLIC po słowie DROP.

Zanalizujmy sekwencję operacji z bieżącego slajdu. Użytkownik OLEK nadał użytkownikowi ALA prawo odczytu i wstawiania nowych rekordów do relacji PRACOWNICY, której jest właścicielem. Użytkownik ALA wykorzystuje to prawo i odczytuje wszystkie rekordy z relacji PRACOWNICY w schemacie użytkownika OLEK. Musi w tym celu przed nazwą relacji podać nazwę schematu. Chcąc uprościć wydawanie zapytań użytkownik ALA tworzy sobie synonim PRAC_OLEK, wskazujący na relację PRACOWNICY w schemacie użytkownika OLEK. Od tej pory w zapytaniach wykorzystuje utworzony synonim.



W bieżącym ćwiczeniu przedstawiono uwierzytelnianie i autoryzację użytkownika bazy danych. Omówiono różne rodzaje uwierzytelniania: przez bazę danych, system operacyjny, usługę sieciową, uwierzytelnianie wielowarstwowe. Dalej przedstawiono mechanizmy, wykorzystywane przy autoryzacji użytkowników: przywileje systemowe, przywileje obiektowe oraz role. Ćwiczenie zakończono prezentacją synonimów jako mechanizmu ułatwiającego odwołania do obiektów w bazie danych.

Każde z omówionych w ćwiczeniu zagadnień zostało utrwalone przez serię zadań.