



**POLITECHNIKA LUBELSKA
WYDZIAŁ ELEKTROTECHNIKI
I INFORMATYKI**

**KIERUNEK STUDIÓW
INFORMATYKA**

Przedmiot: Wprowadzenie do systemów baz danych

Raport z wykonania projektu pt.

**Baza danych wykorzystywana w systemie wspomagania
sklepu stacjonarnego oraz internetowego oferującego
produkty marki Apple (cortland.pl, idream.pl)**

Autor:

Grzegorz Rogowski

Michał Raczyński

Ewelina Plak

Michał Reszka

Mateusz Pielak

Piotr Plewka



Fundusze Europejskie
Wiedza Edukacja Rozwój



**Rzeczpospolita
Polska**

Unia Europejska
Europejski Fundusz Społeczny



Lublin, 2023

ROZDZIAŁ 1. OPIS WYBRANEGO OBSZARU RZECZYWISTOŚCI ORAZ WSKAZANIE PROBLEMU, KTÓRY ZOSTANIE ROZWIĄZANY PRZY POMOCY SYSTEMU INFORMATYCZNEGO WYKORZYSTUJĄCEGO PROJEKTOWANĄ BAZĘ DANYCH

Dokonaj ogólnej prezentacji wybranej rzeczywistości, wskazując procesy w niej realizowane oraz obiekty uczestniczące w tych procesach. Określ logiczną kolejność realizacji tych procesów.

Określ problem lub problemy jakie należy rozwiązać w wybranym obszarze rzeczywistości poprzez wdrożenie systemu informatycznego, w którym zastosujesz projektowaną bazę danych.

- Głównym celem jest rozwiązanie problemu projektowania bazy danych, która zostanie wykorzystana w systemie wspomagania sprzedaży produktów marki Apple w sklepie stacjonarnym jak i internetowym

Kroki:

1. Sformułowanie problemu. Należy podjąć decyzję, jakie informacje będą przechowywane w bazie danych.
2. Należy określić dostępność do bazy danych na różnych poziomach. System posiada funkcjonalności z pozycji klienta, obsługę klienta, administratora oraz pracownika.
3. Należy opisać każdą tabelę, pola, rodzaje zawieranych informacji.
4. Należy opisać każdą relację między tabelami.

Użytkownik klient może zobaczyć własną historię zakupów, liczbę punktów karty lojalnościowej (jeżeli takową posiada, jeśli nie, może taką założyć), aktualny stan koszyka zakupów internetowych, lokalizację sklepów w której dokonał zakupów poprzez podanie karty lojalnościowej. Użytkownik pracownik posiada o wiele więcej funkcji, niż klient. Pracownik przede wszystkim może zamówić produkt i wprowadzić go na stan magazynowy po przyjęciu towaru i ustalić jego cenę. Użytkownik Administrator może wprowadzać promocję na dane produkty i na dany okres trwania promocji. Użytkownik odpowiedzialny za obsługę klienta może przyjmować reklamacje i zarządzać nimi w formie zwrotu pieniędzy, wymiany towaru bądź odesłanie do serwisu.

ROZDZIAŁ 2. SFORMUŁOWANIE CELU BUDOWY SYSTEMU INFORMATYCZNEGO, WYKORZYSTUJĄCEGO PROJEKTOWANĄ BAZĘ DANYCH, ORAZ PREZENTACJA WYMAGAŃ FUNKCYJONALNYCH STAWIANYCH SYSTEMOWI

Określ cel budowy systemu informatycznego, wykorzystującego projektowaną bazę danych. W definicji celu pamiętaj o zastosowaniu zasady SMART, tj. zapewnij, aby zdefiniowany cel był skonkretyzowany, mierzalny, osiągalny, istotny i określony w czasie.

Przedstaw wymagania funkcjonalne jakie powinny być spełnione przez projektowany system IT, aby możliwe było osiągnięcie celu jego budowy. Jeśli system ma być wykorzystywany przez różnego rodzaju użytkowników (np. klient, pracownik, administrator systemu), dokonaj podziału tych wymagań uwzględniając specyficzne potrzeby każdego z nich.

Celem budowy systemu informatycznego, wykorzystującą projektowaną bazę danych jest:

- a) umożliwienie sprzedaży poprzez kanały sprzedaży internetowej,
- b) uproszczenie zarządzania stanem magazynowym sklepu,
- c) uproszczenie obsługi pozakupowej,
- d) uproszczenie zarządzania promocjami i ofertami specjalnymi,
- e) umożliwienie prowadzenia programu lojalnościowego

System jest wykorzystywany przez użytkowników o różnych rolach - klienta, pracownika sklepu, obsługi klienta, administratora.

Użytkownik o roli klienta powinien mieć możliwość:

- a) przeglądania oferty sklepu,
- b) przeglądania produktów w oparciu o wybór kategorii/podkategorii,
- c) przeglądania ofert specjalnych,
- d) dodawania/usuwania produktów do koszyka,
- e) zastosowania kodu rabatowego,
- f) zarejestrowania swojego konta w programie lojalnościowym,
- g) podglądu statusu swojego konta w programie lojalnościowym,
- h) utworzenia zamówienia,
- i) wprowadzenia adresu dostawy,
- j) wyboru metody dostawy,
- k) wyboru metody płatności,
- l) dobrania wysokości obowiązującego podatku,
- m) podglądu statusu zamówienia,
- n) anulowania zamówienia,
- o) wprowadzenia kodu rabatowego przy tworzeniu zamówienia,
- p) zgłoszenia problemów technicznych ze stroną lub zamówionym produktem,
- q) wprowadzenia i edycji danych przypisanych do swojego konta (imię, nazwisko, nr telefonu itd.)

Użytkownik o roli pracownika sklepu powinien mieć możliwość:

- a) aktualizacji stanu magazynowego,
- b) aktualizacji statusu zamówień klientów (w domyśle wydawanie zamówień),
- c) tworzenia zamówienia/sprzedaży produktu,
- d) przyjmowania zgłoszeń reklamacji

Użytkownik o roli obsługi klienta powinien mieć możliwość:

- a) zarządzania zgłoszonymi reklamacjami,
- b) monitorowania i edycji zamówień,
- c) podglądu statusu kont klientów w programie lojalnościowym,
- d) dodawania nowych kart lojalnościowych klientów,
- e) kontaktu z użytkownikami zgłaszającymi problemy techniczne,
- f) zastosowania zmian do danych przypisanych do konta klienta.

Użytkownik o roli administratora powinien mieć możliwość:

- a) monitorowania działań pracowników i obsługi klienta,
- b) dodawania produktów i kategorii/podkategorii produktowych,
- c) zarządzania promocjami,
- d) tworzenia kodów promocyjnych,
- e) przyznawania/odbierania odpowiednich ról poszczególnym kontom (domyślnie tworzone konta posiadają uprawnienia klienta).



ROZDZIAŁ 3. SZCZEGÓŁOWY OPIS PROCESÓW REALIZOWANYCH W WYBRANEJ RZECZYWISTOŚCI, ZALEŻNOŚCI MIĘDZY NIMI ORAZ OBIEKTÓW W NICH UCZESTNICZĄCYCH

Przedstaw szczegółowo procesy jakie będą zachodzić w wybranej rzeczywistości. Jeżeli dany proces ma charakter złożony, dokonaj jego podziału na działania (akcje) elementarne oraz opisz kolejność ich realizacji.

Opisz zależności pomiędzy poszczególnymi procesami, wskazując logiczną kolejność ich realizacji.

Wskaż (nazwij) obiekty, które będą uczestniczyć w realizacji poszczególnych procesów.

- Klient:

- A. Proces zakładania nowego konta

- a. Należy podać email, telefon, kraj, region, miasto, kod pocztowy i ulicę. Dane są kierowane do tabeli User.
- b. Podany kraj jest kierowany do tabeli Country, podane miasto jest kierowane do tabeli City
- c. założenie konta należy potwierdzić odpowiedzią na wiadomość wysłaną na podany email

- B. Proces logowania się na konto - wymagane jest podanie e-mail/nr telefonu jako login konta

- C. Proces edycji konta

- a. zmiany powodują zmiany danych w atrybutach tabeli User.
- b. jeżeli jest to konieczne, wszelkie zmiany należy uwzględnić w pozostałych tabelach.
- c. potwierdzenie zmian zachodzi przez wiadomość wysłaną na podany email.

- D. Proces zarządzania produktami w koszyku - lista przedmiotów w koszyku musi być aktualizowana z każdym dodaniem lub usunięciem wybranego produktu.

- E. Proces tworzenia zamówienia

- a. zamówienie jest tworzone na podstawie koszyka zakupowego, z uwzględnieniem ilości wybranych produktów. szczegóły zamówionych produktów są pobierane z tabeli OrderItem i przechowywane w atrybutach Amount oraz TotalAmount. Uwzględniona jest także informacja o podatkach, przechowywana w atrybucie Tax.
- b. wybierany jest adres na podstawie danych z tabeli user lub danych wybranych ręcznie. szczegóły kierowane są do atrybutu Address z tabeli Order.
- c. wybierana jest metoda płatności z tabeli PaymentMethod, wybór jest przechowywany w atrybucie PaymentMethod w tabeli Order.
- d. możliwe jest wpisanie kuponu który umożliwia nabycie wybranych produktów po obniżonej cenie lub obniżenia wartości zamówienia.
- e. po podsumowaniu szczegółów zamówienia i jego potwierdzeniu zamówienie dostaje status i unikalne ID.
- f. zamówienie może zostać anulowane poprzez kontakt z pracownikiem sklepu.

- F. Proces składania reklamacji

- a. reklamacja jest kierowana do pracownika.



Zintegrowany Program Rozwoju Politechniki Lubelskiej – część druga

- b. podawane jest ID zamówienia i informacje o produktach poddawanych reklamacji
- c. na mocy decyzji pracownika, zweryfikowanej przez pracownika z działu obsługi klienta, reklamacja jest wykonana poprzez wymianę produktu bądź zwrot pieniędzy za dany produkt.
- G. zgłaszanie problemów technicznych- klient może zgłosić występowanie problemów technicznych do działu obsługi klienta
- H. składanie wniosku o założenie karty lojalnościowej- klient może złożyć wniosek o wybraną kartę lojalnościową do działu obsługi klienta.
- Pracownik:
 - A. proces zarządzania stanem magazynowym
 - a. zmiany są wprowadzane na podstawie operacji zachodzących w sklepie (zamówienia, zwroty, dostawy)
 - b. wszelkie zmiany w składzie są aktualizowane w tabeli ShopSupply
 - B. proces zarządzania zamówieniami
 - a. zamówienie jest realizowane na podstawie jego szczegółów: produktów i ich ilości
 - b. po skompletowaniu zamówienia i wysłaniu do kuriera, należy zmienić status zamówienia
 - c. kolejne zmiany statusu są wprowadzane na podstawie procesów na etapie kurier-klient.
 - C. Proces zarządzania reklamacjami
 - a. przyjęta reklamacja musi być rozpatrzona na podstawie informacji podanych przez reklamującego.
 - b. jeżeli uzasadnienie reklamacji umożliwia jej przyjęcie, pracownik podejmuje dalsze działania: przyjęcie zwrotu i wybrana forma przekazania środków do klienta: zwrot pieniędzy bądź wysyłka zastępczego produktu. Dokonany proces reklamacji jest notowany w tabelach Refunds oraz RefundsItem.
- Obsługa Klienta:
 - A. zarządzanie reklamacjami- pracownik działu obsługi klienta weryfikuje reklamację i podejmuje decyzje o dalsze działania
 - B. monitorowanie i edycja zamówień
 - a. zamówienie które jest w trakcie kompletowania może być edytowane na życzenie klienta: modyfikacja szczegółów zamówienia bądź jego anulowanie.
 - b. przebieg poszczególnych etapów zamówienia jest monitorowany i tej podstawie dokonywane są aktualizacje statusu.
 - C. monitorowanie statusów kont lojalnościowych klientów
 - D. nadawanie kart lojalnościowych- na podstawie wniosku klienta, pracownik działu obsługi klienta może dodać daną kartę lojalnościową do konta wnioskodawcy. Informacje o karcie i członkostwie są notowane w tabeli MemberClub.



Zintegrowany Program Rozwoju Politechniki Lubelskiej – część druga

- E. kontakt z klientami zgłaszającymi problemy techniczne- zgłoszenia problemów technicznych są weryfikowane i w wypadku ich zatwierdzenia- są przekazywane do odpowiednich pracowników.
- F. Weryfikacja edycji informacji kont użytkowników- po wprowadzeniu zmian i zatwierdzeniu ich drogą mailową przez klienta, pracownik edytuje informacje danego konta. Zmiany zachodzą w tabeli User i w razie konieczności zachodzą w innych tabelach (np. w tabeli City w przypadku zmiany informacji o mieście).
- Administrator:
 - A. proces dodawanych kategorii/podkategorii produktowych
 - B. zarządzanie cennikiem i promocjami
 - a. podczas edycji cennika, administrator określa cenę produktu , okres ważności danej ceny oraz ilość dostępnych produktów.
 - b. wprowadzane dane są zapisywane w tabelach Pricebook i PricebookEntry.
 - c. administrator tworząc promocję, określa typ, wartość, okres ważności i zakres (które produkty są objęte promocją).
 - d. podane informacje są kierowane do tabeli Pricebook z wyjątkiem zakresu, który jest kierowany do PricebookEntry (atrybut Product).
 - e. administrator może nałożyć ograniczenie na dostępną ilość sztuk produktu (atrybut Amount w tabeli PricebookEntry).
 - C. zarządzanie rolami użytkowników- administrator może modyfikować role wybranych użytkowników a także zarządzać statusami kont. Powoduje to zmiany w atrybutach Role i Status w tabeli User.

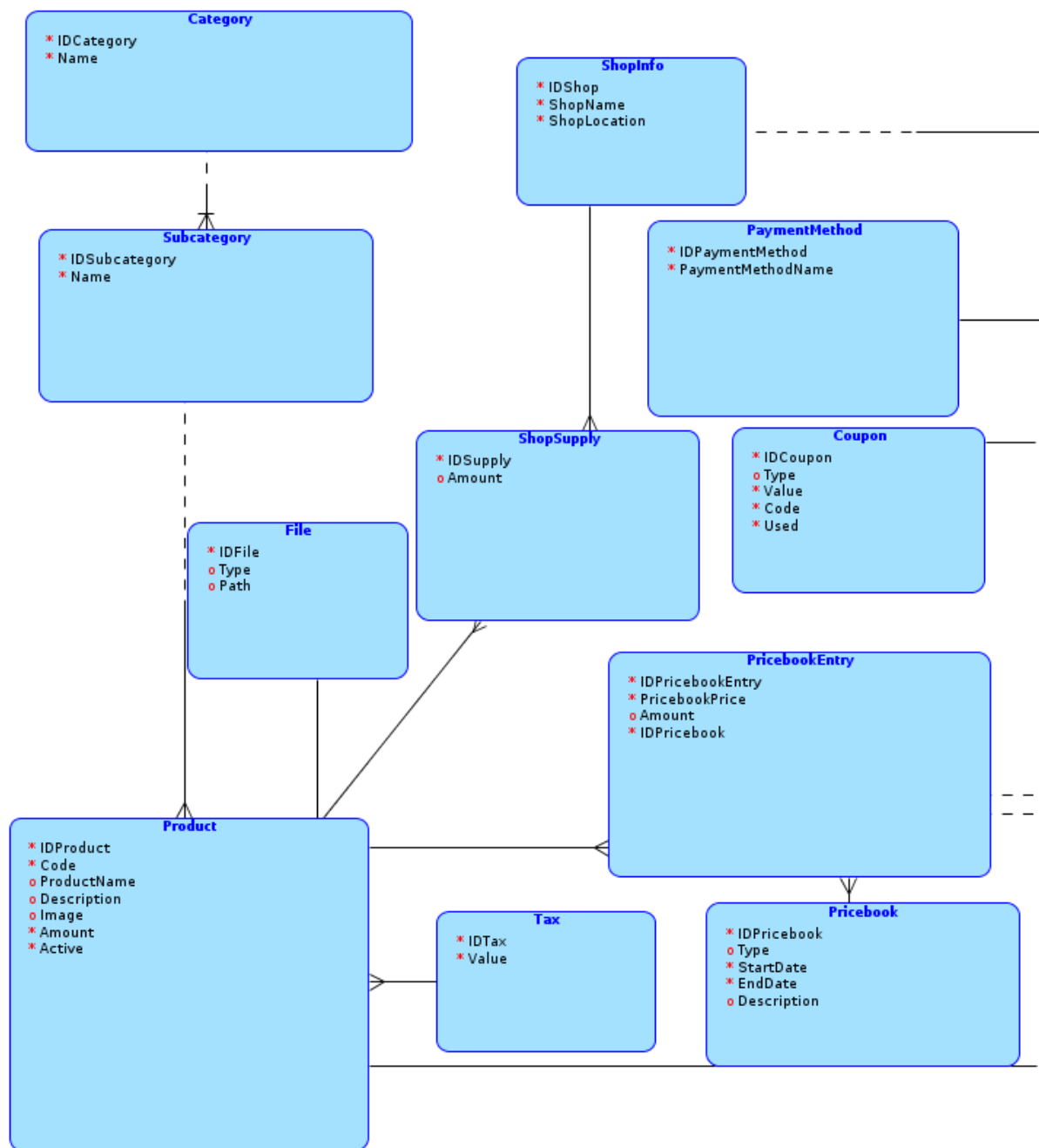


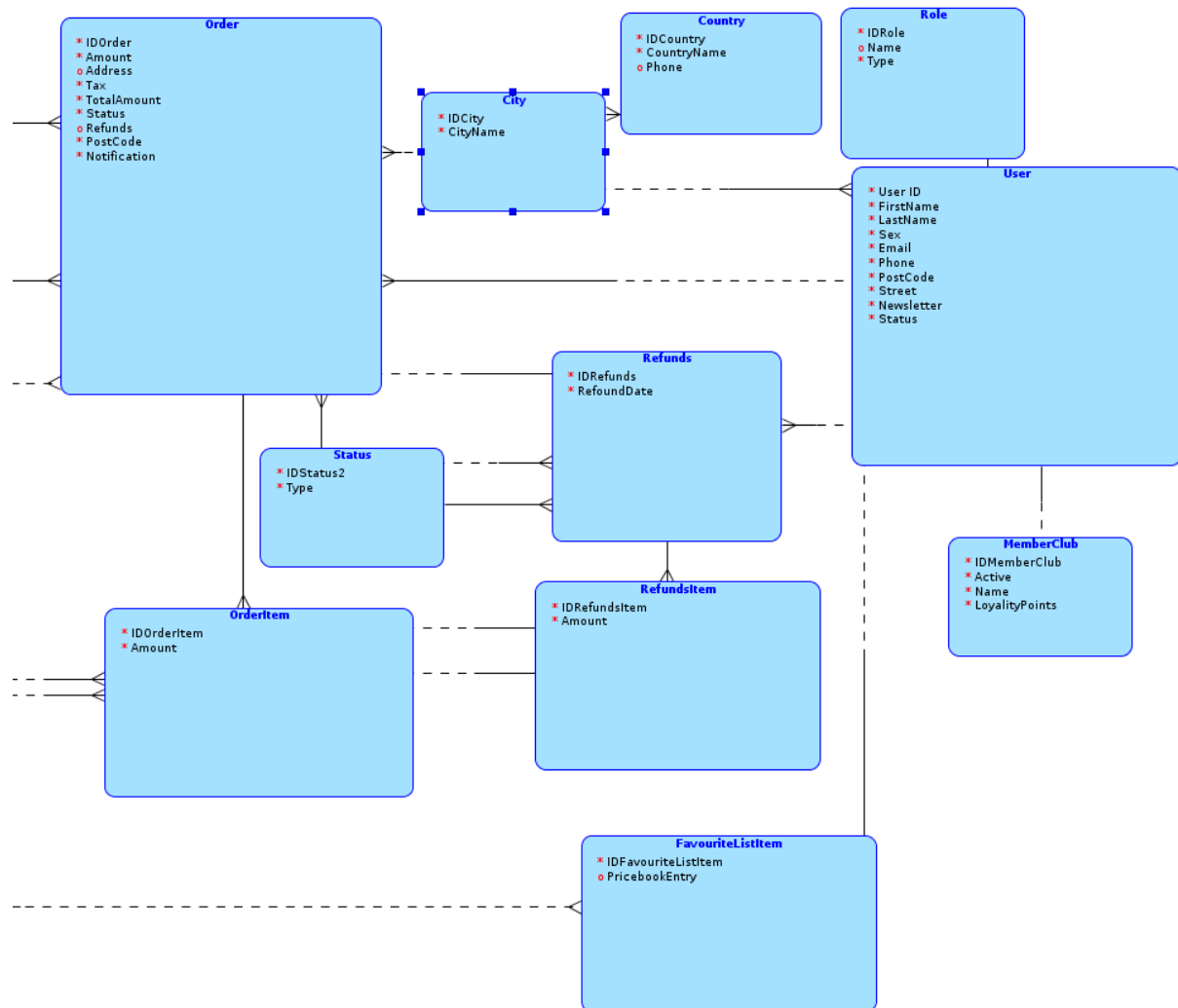
ROZDZIAŁ 4. MODEL KONCEPTUALNY PROJEKTOWANEJ BAZY DANYCH

Przedstaw w formie graficznej model konceptualny projektowanej bazy danych.

Wymień i scharakteryzuj encje, które będą uwzględnione w modelowanej bazie danych. Podaj ich nazwy oraz opisz ogólnie, jakie obiekty rzeczywistości będą one opisywały. Opisz rodzaje związków pomiędzy poszczególnymi encjami.

Model konceptualny:





Nazwa encji:	Opis encji:
User	Informacje o użytkowniku posiadającym zarejestrowane konto w sklepie internetowym.
Role	Informacje o roli przypisanej do konta użytkownika, określające dostęp do poszczególnych funkcji systemu.
Refunds	Informacje o zwrotach zamówień przekazanych za pośrednictwem formularza/zgłoszenia mailowego/telefonicznego
RefundsItem	Dokładne informacje o zwrocie zawierające dane zamówienia oraz datę zwrotu
Country	Informacje o krajach, w których usługi sklepu są realizowane, a także ich numery kierunkowe.
City	Informacje o identyfikatorach użytkowników korzystających z usług w danym mieście oraz zamówieniach do nich dostarczonych.
Order	Informacje o złożonym zamówieniu. Opisywana encja zawiera informacje o promocji, użytkownika, kuponach rabatowych, ilości zamówienia, adresie dostarczenia, metodzie płatności, należności podatkowej, podsumowaniu zamówienia, statusie, zwrotach oraz newslettera.
OrderItem	Informacje o zamówieniu, cenniku, jakości oraz koszcie produkcyjnym.
PaymentMethod	Informacje o metodzie płatności wybranej dla danego zamówienia.
Coupon	Informacje o typie kodu rabatowego, wysokości rabatu, numerze kodu oraz produkcie, którego dotyczy.



Zintegrowany Program Rozwoju Politechniki Lubelskiej – część druga

Status	Informacje o statusie zamówienia - przyjęty do realizacji/zatwierdzono realizację/zrealizowano/produkt gotowy do wysyłki/wysłano/dostarczone.
Product	Informacje o produkcie
File	Informacje o pliku zawierającym opis produktu.
Tax	Informacje o podatku obowiązującym dla danego produktu.
Category	Informacje o kategorii produktowej.
Subcategory	Informacje o podkategorii produktowej.
PricebookEntry	Informacje o danym cenniku.
Pricebook	Informacje o cenie obowiązującej dla danego produktu.
FavouriteListItem	Informacje o produktach dodanych do ulubionych podczas przeglądania ofert w sklepie internetowym przez klienta.
MemberClub	Informacje o członkostwie klienta w programie lojalnościowym.
ShopInfo	Informacje o nazwie i lokalizacji sklepu
ShopSupply	Informacje o zaopatrzeniu danego sklepu



Fundusze Europejskie
Wiedza Edukacja Rozwój



**Rzeczpospolita
Polska**

Unia Europejska
Europejski Fundusz Społeczny



Zintegrowany Program Rozwoju Politechniki Lubelskiej – część druga
Rodzaje związków pomiędzy poszczególnymi encjami:

Encja źródłowa:	Encja docelowa:	Rodzaj związku:
Tax	Product	1:N
Product	PricebookEntry	1:N
Pricebook	PricebookEntry	1:N
Product	FavouriteListItem	1:N
Product	File	1:1
Product	ShopSupply	1:N
ShopInfo	ShopSupply	1:N
Category	Subcategory	1:N
Subcategory	Product	1:N
PricebookEntry	OrderItem	1:N
Order	OrderItem	1:N
Order	Refunds	1:1
Status	Refunds	1:N
PaymentMethod	Order	1:N
ShopInfo	Order	1:N
Status	Order	1:N
City	Order	1:N
Coupon	Order	1:N
User	Order	1:N
User	Refunds	1:N
User	FavouriteListItem	1:1
Refunds	RefundsItem	1:N
OrderItem	RefundsItem	1:1
City	User	1:N



Fundusze Europejskie
Wiedza Edukacja Rozwój



**Rzeczpospolita
Polska**

Unia Europejska
Europejski Fundusz Społeczny



Zintegrowany Program Rozwoju Politechniki Lubelskiej – część druga

MemberClub	User	1:1
Role	User	1:1
Country	City	1:N



Fundusze Europejskie
Wiedza Edukacja Rozwój



**Rzeczpospolita
Polska**

Unia Europejska
Europejski Fundusz Społeczny



ROZDZIAŁ 5. MODEL ZWIĄZKÓW ENCJI PROJEKTOWANEJ BAZY DANYCH

Przedstaw w formie graficznej model logiczny (związków encji) projektowanej bazy danych.

Wymień i opisz znaczenie atrybutów dla poszczególnych encji występujących w prezentowanym modelu logicznym bazy danych. Podaj nazwę i rodzaj każdego atrybutu encji.

Określ dla każdego atrybutu encji jego dziedzinę, maksymalny rozmiar danych, opcjonalność lub konieczność wystąpienia wartości określonej oraz inne ograniczenia integralnościowe, np. unikalność wartości atrybutu bądź też zawężenie jego dziedziny.

Dokonaj szczegółowego opisu każdego związku pomiędzy poszczególnymi encjami.

Tax	Typ danych	Unikalność	Obowiązkowość	Rodzaj
IDTax	Integer(10)	Tak	Tak	Klucz główny
Value	Integer(2)	Nie	Tak	Wartość podatku

Category	Typ danych	Unikalność	Obowiązkowość	Rodzaj
IDCategory	Integer	Tak	Tak	Klucz główny
Name	Varchar(20)	Tak	Tak	Nazwa kategorii

City	Typ danych	Unikalność	Obowiązkowość	Rodzaj
IDCity	Integer(10)	Tak	Tak	Klucz główny
CityName	Varchar(20)	Nie	tak	Nazwa miasta

Country	Typ danych	Unikalność	Obowiązkowość	Rodzaj
Phone	Integer(15)	Tak	Nie	Nr. tel. użytkownika
CountryName	Varchar(25)	Tak	Tak	Nazwa kraju
IDCountry	Integer(10)	Tak	Tak	Klucz główny



Zintegrowany Program Rozwoju Politechniki Lubelskiej – część druga

Coupon	Typ danych	Unikalność	Obowiązkowość	Rodzaj
IDCoupon	Integer(10)	Tak	Tak	Klucz główny
Type	Varchar(15)	Nie	Nie	Rodzaj kuponu
Value	Integer(2)	Nie	Tak	Wartość procentowa
Code	Varchar(10)	Nie	Tak	Nazwa kodu
Used	Boolean	Nie	Tak	1/0, aktywność kuponu

MemberClub	Typ danych	Unikalność	Obowiązkowość	Rodzaj
IDMemberClub	Integer(10)	Tak	Tak	Klucz główny
Name	Varchar(20)	Tak	Tak	Nazwa programu lojalnościowego
Active	Boolean	Nie	Tak	1 lub 0, określa aktywność programu
LoyaltyPoints	Numeric(5)	Nie	Tak	Liczba punktów na karcie lojalnościowej

Zintegrowany Program Rozwoju Politechniki Lubelskiej – część druga

Order	Typ danych	Unikalność	Obowiązkowość	Rodzaj
IDOrder	Integer(10)	Tak	tak	Klucz zamówienia
Amount	Integer(10)	nie	tak	Liczba artykułów w zamówieniu
Address	Varchar	nie	nie	Adres zamówienia lub numer paczkomatu, dla zamówienia stacjonarnego brak
PaymentMethod	Varchar(7)	nie	tak	Metoda płatności
Tax	Integer(2)	nie	tak	Kwota podatku podana w %
TotalAmount	Float(10)	nie	tak	Całkowita kwota do zapłaty wraz z kosztem dostawy
Status	Varchar(15)	nie	tak	Status zamówienia
Refunds	Boolean	nie	nie	1/0, określa czy zamówienie zostało zwrócone
Notification	Varchar(15)	nie	tak	Powiadomienie o stanie zamówienia

OrderItem	Typ danych	Unikalność	Obowiązkowość	Rodzaj
IDOrderItem	Integer(10)	Tak	Tak	klucz główny
Amount	Integer(10)	Nie	Tak	Ilość konkretnych produktów w zamówieniu

PaymentMethod	Typ danych	Unikalność	Obowiązkowość	Rodzaj
IDPaymentMethod	Integer(10)	tak	tak	Klucz główny
PaymentMethodName	Varchar(20)	nie	tak	Nazwa metody płatności

Zintegrowany Program Rozwoju Politechniki Lubelskiej – część druga

Product	Typ danych	Unikalność	Obowiązkowość	Rodzaj
IDProduct	Integer(10)	tak	tak	Klucz główny
Code	Varchar(20)	nie	tak	Kod producenta dla produktu
Description	Varchar(50)	nie	nie	Opis produktu
Image	Image	nie	nie	Zdjęcie produktu
Amount	Integer(10)	nie	tak	
Active	Boolean	nie	tak	1/0, określa dostępność produktu do sprzedaży
ProductName	Integer(20)	nie	nie	Nazwa produktu

Pricebook	Typ danych	Unikalność	Obowiązkowość	Rodzaj
IDPricebook	Integer(10)	tak	tak	Klucz główny
Type	Varchar(15)	nie	tak	Typ ceny, np. stała, promocyjna
StartDate	Datetime	nie	tak	Od kiedy obowiązuje cena
EndDate	Datetime	nie	tak	Do kiedy
Description	Varchar(50)	nie	nie	Opis promocji, zmiany ceny

PricebookEntry	Typ danych	Unikalność	Obowiązkowość	Rodzaj
IDPricebookEntry	Integer(10)	tak	tak	Klucz główny
PricebookPrice	Float(10)	nie	tak	Faktyczna cena produktu w danej chwili
Amount	Numeric(10)	nie	nie	Liczba produktów w danej promocji

Zintegrowany Program Rozwoju Politechniki Lubelskiej – część druga

Refunds	Typ danych	Unikalność	Obowiązkowość	Rodzaj
IDRefund	Integer(10)	tak	tak	Klucz główny
RefoundDate	Datetime	nie	ttak	Data zwrotu produkt

RefundsItem	Typ danych	Unikalność	Obowiązkowość	Rodzaj
IDRefundItem	Integer(10)	tak	tak	Klucz główny
Amount	Integer(10)	nie	tak	Liczba produktów w zwrocie

Role	Typ danych	Unikalność	Obowiązkowość	Rodzaj
IDRole	Integer(10)	tak	tak	Klucz główny
Name	Varchar(20)	tak	nie	Nazwa roli w systemie

ShopInfo	Typ danych	Unikalność	Obowiązkowość	Rodzaj
IDShop	Integer(10)	tak	tak	Klucz główny
ShopName	Varchar(20)	nie	tak	Nazwa sklepu stacjonarnego + miasto i dzielnica
ShopLocation	Varchar(50)	tak	tak	Pełny adres(miasto, ulica, kod pocztowy)

ShopSupply	Typ danych	Unikalność	Obowiązkowość	Rodzaj
IDSupply	Integer(10)	tak	tak	Klucz główny
Amount	Integer(10000)	nie	tak	Zaopatrzenie magazynowe danego sklepu oraz magazynu wysyłkowego

Zintegrowany Program Rozwoju Politechniki Lubelskiej – część druga

Status	Typ danych	Unikalność	Obowiązkowość	Rodzaj
IDStatus	Integer(10)	tak	tak	Klucz główny
Status	Varchar(20)	nie	tak	Status zamówienia lub zwrotu

Subcategory	Typ danych	Unikalność	Obowiązkowość	Rodzaj
IDSubcategory	Integer(10)	tak	tak	Klucz główny
Name	Varchar(20)	nie	tak	Nazwa podkategorii produktów

User	Typ danych	Unikalność	Obowiązkowość	Rodzaj
IDUser	Integer(10)	tak	tak	Klucz główny
FirstName	Varchar(20)	nie	tak	Imię
LastName	Varchar(20)	nie	tak	Nazwisko
Sex	Varchar(1)	nie	tak	Płeć
Email	Varchar(30)	tak	tak	Email użytkownika
Phone	Integer(15)	tak	tak	Telefon komórkowy
PostCode	Varchar(11)	nie	tak	Kod pocztowy
Street	Varchar(25)	nie	tak	Ulica
Newsletter	boolean	nie	tak	Czy user należy do newslettera
Status	boolean	nie	tak	Aktywność konta

Zintegrowany Program Rozwoju Politechniki Lubelskiej – część druga

File	Typ danych	Unikalność	Obowiązkowość	Rodzaj
IDFile	Integer(10)	tak	tak	Klucz główny
Type	PDF/jpg	nie	nie	Typ pliku
Path	Varchar(50)	tak	nie	Ścieżka pliku



Fundusze Europejskie
Wiedza Edukacja Rozwój



**Rzeczpospolita
Polska**

Unia Europejska
Europejski Fundusz Społeczny



Szczegółowy opis każdego związku

User - Role
Do jednego „User’a” może zostać przypisanych wiele ról (Role), a jeden „User”. Związek jest 1:N i jest obowiązkowy.
Country - City
Jeden „Country” może należeć do wielu „City”, a z jednego „Country” może pochodzić jeden „City”. Związek jest 1: N i jest obowiązkowy.
User - Refunds
Jeden „User” może wykonać wiele zwrotów (Refunds), a wiele „Refunds” może pochodzić od jednego „User’a”. Związek jest 1:N i jest obowiązkowy.
User - Order
Jeden „User” może wykonać wiele zamówień (Order), a z wiele „Order” może pochodzić od jednego „User’a”. Związek jest 1:N i jest obowiązkowy.

Tax - Product
Jeden „Tax” może być zawarty w wielu produktach (Product), a z wiele „Product” może mieć jeden „Tax”. Związek jest 1:N i jest obowiązkowy.



Product - PricebookEntry
Jeden „Product” może łączyć się z wieloma „PricebookEntry”, a wiele „PricebookEntry” może zawierać jeden „Product”. Związek jest 1:N i jest obowiązkowy.
Product - FavouriteListItem
Jeden „Product” może być dodany wiele razy do „FavouriteListItem”, a wiele „FavouriteListItem” może zawierać ten sam „Product”. Związek jest 1:N i jest obowiązkowy.
Product - File
Jeden „Product” może zawierać opis produktu „File”, a jeden „File” może opisywać „Product”. Związek jest 1:1 i jest obowiązkowy.

Pricebook - PricebookEntry
Jedna „Pricebook” może być zawarty w wielu wpisach promocyjnych (PricebookEntry), a z wiele „PricebookEntry” może mieć jedną „Pricebook”. Związek jest 1:N i jest obowiązkowy.

Category - SubCategory
Jedna „Category” może mieć wiele podkategorii (SubCategory), a z wiele „SubCategory” może należeć do jednej „Category”. Związek jest 1:N i jest obowiązkowy.

SubCategory - Product
Jedna „SubCategory” może mieć wiele produktów (Product), a z wiele „Product” może należeć do jednej „SubCategory”. Związek jest 1:N i jest obowiązkowy.

Order - OrderItem
Jedno „Order” może składać się z wielu elementów „OrderItem”, a wiele „OrderItem” może zawierać jedno „Order”. Związek jest 1:N i jest obowiązkowy.
Order - PaymentMethod
Jedno „Order” może proponować wiele metod płatności „PaymentMethod”, a wiele „PaymentMethod” może opłacić jedno „Order”. Związek jest 1:N i jest obowiązkowy.



Order - City

Jeden „Order” może zostać zrealizowane w jednym mieście „City”, a jeden „City” może zrealizować jeden „Order”. Związek jest 1:1 i jest obowiązkowy.

Coupon - Order

Jeden „Coupon” może zostać użyty przy wielu produktach (Product), a wiele „Product” może obsługiwać realizację jednego „Coupon”. Związek jest 1:N i jest obowiązkowy.

Refunds - RefundItem

Jeden „Refund” może mieć jeden element do zwrotu „RefundItem”, a jeden „RefundItem” może zostać przyjęty jako „Refund”. Związek jest 1:1 i jest obowiązkowy.

FavouriteListItem - User

Jedna „FavouriteListItem” może mieć wielu „User”, a wielu „User” może mieć jedną „FavouriteListItem”. Związek jest 1:N i jest obowiązkowy.

City - User

Jedno „City” może mieć wielu „User”, a wielu „User” może należeć do jednego „City”. Związek jest 1:N i jest obowiązkowy.

MemberClub - User

Jedno „MemberClub” może mieć wielu „User”, a wielu „User” może należeć do jednego „MemberClub”. Związek jest 1:N i jest obowiązkowy.

ShopInfo - ShopSupply

Jedno „ShopInfo” może mieć wiele „ShopSupply”, a wiele „ShopSupply” może należeć do jednego „ShopInfo”. Związek jest 1:N i jest obowiązkowy.

ShopInfo - Order

Jedno „ShopInfo” może mieć wiele „Order”, a wiele „Order” może należeć do jednego



„ShopInfo”. Związek jest 1:N i jest obowiązkowy.

ShopSupply - Product

Jedno „Product” może mieć wiele „ShopSupply”, a wiele „ShopSupply” może należeć do jednego „Product”. Związek jest 1:N i jest obowiązkowy.

Status - Order

Jedno „Status” może mieć wiele „Order”, a wiele „Order” może należeć do jednego „Status”. Związek jest 1:N i jest obowiązkowy.

Status - Refunds

Jedno „Status” może mieć wiele „Refunds”, a wiele „Refunds” może należeć do jednego „Status”. Związek jest 1:N i jest obowiązkowy.

Order - Refunds

Jedno „Order” może mieć jeden zwrot „Refunds”, a jeden „Refunds” może dotyczyć jednego „Order”. Związek jest 1:1 i jest obowiązkowy.

PricebookEntry - OrderItem

Wiele „OrderItem” może należeć do jednego „PricebookEntry”, a jeden „PricebookEntry” może dotyczyć wielu „OrderItem”. Związek jest 1:N i jest obowiązkowy.

OrderItem - RefundsItem

Jeden „OrderItem” może należeć do jednego „RefundsItem”, a jeden „RefundsItem” może dotyczyć jednego „OrderItem”. Związek jest 1:1 i jest obowiązkowy.



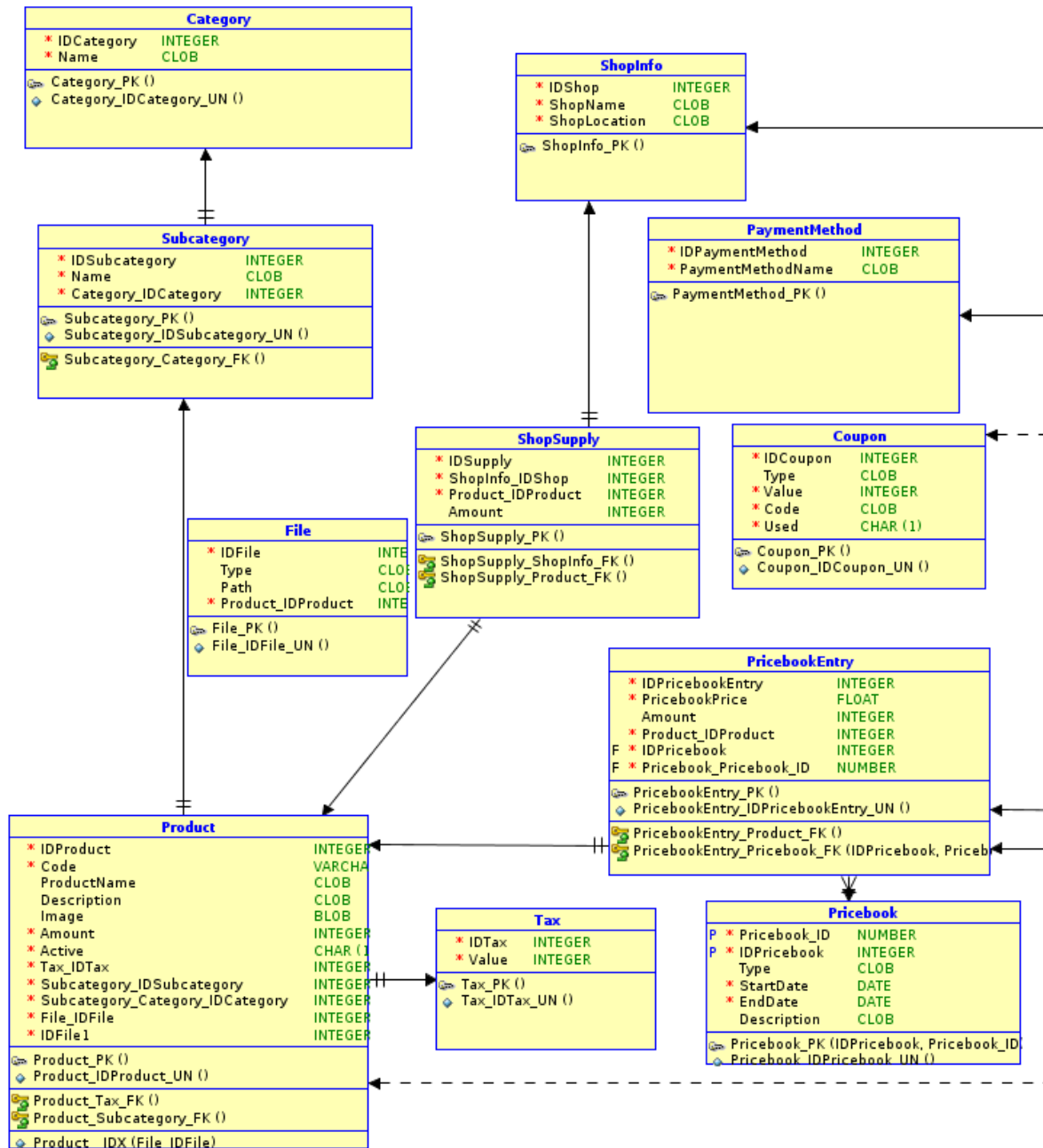
ROZDZIAŁ 6. MODEL RELACYJNY PROJEKTOWANEJ BAZY DANYCH

Przedstaw w formie graficznej model relacyjny projektowanej bazy danych.

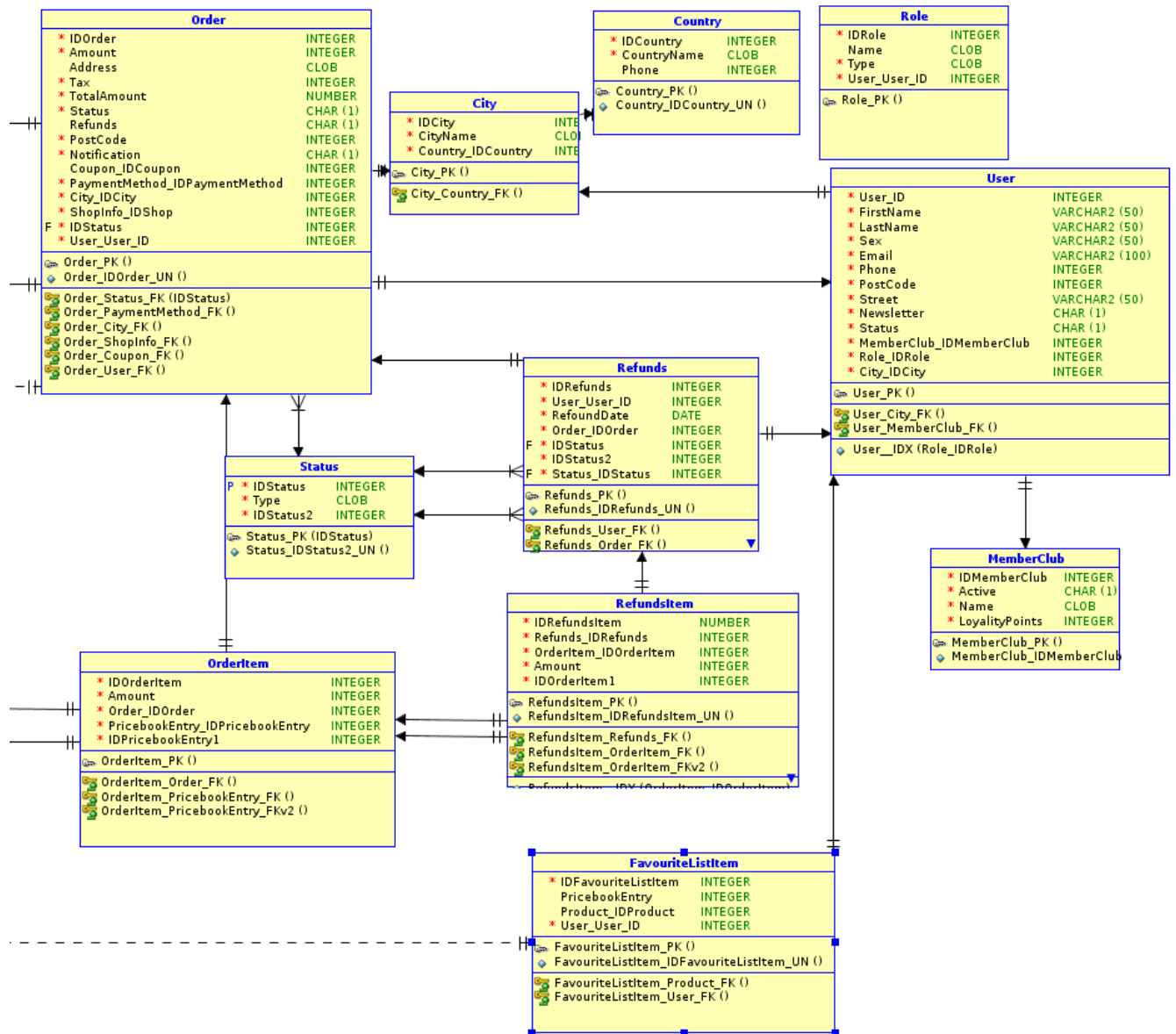
Określ ograniczenia integralnościowe zastosowane dla poszczególnych kolumn poszczególnych tabeli. Podaj nazwy tych ograniczeń oraz ich rodzaj.

Wymień indeksy, jakie należy utworzyć dla poszczególnych tabel. Podaj nazwę indeksu, jego rodzaj oraz kolumny użyte do jego zbudowania.

Zintegrowany Program Rozwoju Politechniki Lubelskiej – część druga



Zintegrowany Program Rozwoju Politechniki Lubelskiej – część druga



Fundusze Europejskie
Wiedza Edukacja Rozwój



Rzeczpospolita Polska

Unia Europejska
Europejski Fundusz Społeczny



Category					
Nazwa kolumny	Nazwa ograniczenia	Opis ograniczenia	PK	FK	Unikalność
IdCategory	INT NOT NULL	kolumna nie może mieć wartości NULL, przyjmowany m typem danych jest INT, czyli liczba całkowita	Tak(category_pk)	Nie	Tak(idcategory_un)
Name	CLOB NOT NULL	przechowuje duże ilości plików typu character, musi zawierać wartość - nie może być puste	Nie	Nie	Nie

City					
Nazwa kolumny	Nazwa ograniczenia	Opis ograniczenia	PK	FK	Unikalność
IdCity	INT NOT NULL	kolumna nie może mieć wartości NULL, przyjmowany m typem danych jest INT, czyli liczba całkowita	Tak(city_pk)	Nie	Nie
CityName	CLOB NOT NULL	przechowuje duże ilości plików typu character, musi zawierać	Nie	Nie	Nie

Zintegrowany Program Rozwoju Politechniki Lubelskiej – część druga

		wartość - nie może być puste			
IdCountry	INT NOT NULL	kolumna nie może mieć wartości NULL, przyjmowany m typem danych jest INT, czyli liczba całkowita	Nie	Nie	Nie

Country					
Nazwa kolumny	Nazwa ograniczenia	Opis ograniczenia	PK	FK	Unikalność
IdCountry	INT NOT NULL	kolumna nie może mieć wartości NULL, przyjmowany m typem danych jest INT, czyli liczba całkowita	Tak(country_pk)	Nie	Tak(idcountry_un)
CountryName	CLOB NOT NULL	przechowuje duże ilości plików typu character, musi zawierać wartość - nie może być puste	Nie	Nie	Nie
Phone	INT	Przyjmowany m typem danych jest INT, czyli liczba całkowita	Nie	Nie	Nie

Coupon					
Nazwa kolumny	Nazwa ograniczenia	Opis ograniczenia	PK	FK	Unikalność
IdCoupon	INT NOT NULL	kolumna nie może mieć wartości NULL, przyjmowany m typem danych jest INT, czyli liczba całkowita	Tak(coupon_pk)	Nie	Tak(idcoupon_un)
CountryName	CLOB NOT NULL	przechowuje duże ilości plików typu character, musi zawierać wartość - nie może być puste	Nie	Nie	Nie
Phone	INT	Przyjmowany m typem danych jest INT, czyli liczba całkowita	Nie	Nie	Nie

FacouriteListItem					
Nazwa kolumny	Nazwa ograniczenia	Opis ograniczenia	PK	FK	Unikalność
IdFacouriteListItem	INT NOT NULL	kolumna nie może mieć wartości NULL, przyjmowany m typem danych jest INT, czyli liczba całkowita	Tak(favourite listitem_pk)	Nie	Tak(idfavourite listitem_un)
PriceBookEntry	INT	Przyjmowany m typem	Nie	Nie	Nie

Zintegrowany Program Rozwoju Politechniki Lubelskiej – część druga

		danych jest INT, czyli liczba całkowita			
IdProduct	INT	Przyjmowany m typem danych jest INT, czyli liczba całkowita	Nie	Tak	Nie
IdUser	INT NOT NULL	kolumna nie może mieć wartości NULL, przyjmowany m typem danych jest INT, czyli liczba całkowita	Nie	Tak	Nie

File					
Nazwa kolumny	Nazwa ograniczenia	Opis ograniczenia	PK	FK	Unikalność
IdFile	INT NOT NULL	kolumna nie może mieć wartości NULL, przyjmowany m typem danych jest INT, czyli liczba całkowita	Tak(ffile_pk)	Nie	Tak(idfile_un)
Type	CLOB	przechowuje duże ilości plików typu character,	Nie	Nie	Nie
Path	CLOB	przechowuje duże ilości plików typu character,	Nie	Tak	Nie

MemberClub					
Nazwa kolumny	Nazwa ograniczenia	Opis ograniczenia	PK	FK	Unikalność
IdMemberClub	NUMBER NOT NULL	kolumna nie może mieć wartości NULL, przyjmowany m typem danych jest NUMBER, czyli można przechowywać liczby całkowite i zmiennoprzecinkowe	Tak(memberclub_pk)	Nie	Tak(idmemberclub_un)
Active	CHAR(1) NOT NULL	przechowuje znaki i długości 1 bajta, nie może przyjmować wartości NULL	Nie	Nie	Nie
Name	CLOB NOT NULL	przechowuje duże ilości plików typu character, musi zawierać wartość - nie może być puste	Nie	Tak	Nie
LoyaltyPoints	INT NOT NULL	kolumna nie może mieć wartości NULL, przyjmowany m typem danych jest INT, czyli liczba całkowita	Nie	Tak	Nie



Order					
Nazwa kolumny	Nazwa ograniczenia	Opis ograniczenia	PK	FK	Unikalność
IdOrder	INT NOT NULL	kolumna nie może mieć wartości NULL, przyjmowany m typem danych jest INT, czyli liczba całkowita	Tak(order_pk)	Nie	Tak(order_id order_un)
Amount	INT NOT NULL	kolumna nie może mieć wartości NULL, przyjmowany m typem danych jest INT, czyli liczba całkowita	Nie	Nie	Nie
Address	CLOB	przechowuje duże ilości plików typu character,	Nie	Nie	Nie
Tax	INT NOT NULL	kolumna nie może mieć wartości NULL, przyjmowany m typem danych jest INT, czyli liczba całkowita	Nie	Tak	Nie
TotalAmount	NUMBER NOT NULL	kolumna nie może mieć wartości NULL, przyjmowany m typem danych jest NUMBER, czyli można	Nie	Nie	Nie

Zintegrowany Program Rozwoju Politechniki Lubelskiej – część druga

		przechowywać liczby całkowite i zmiennoprzecinkowe			
Status	CHAR(1) NOT NULL	przechowuje znaki o długości 1 bajta, nie może przyjmować wartości NULL	Nie	Tak(order_status_fk)	Nie
Refunds	CHAR(1)	przechowuje znaki długości 1 bajta	Nie	Nie	Nie
PostCode	INT NOT NULL	kolumna nie może mieć wartości NULL, przyjmowany m typem danych jest INT, czyli liczba całkowita	Nie	Nie	Nie
Notification	CHAR(1) NOT NULL	przechowuje znaki o długości 1 bajta, nie może przyjmować wartości NULL	Nie	Nie	Nie
IdUser	INT NOT NULL	kolumna nie może mieć wartości NULL, przyjmowany m typem danych jest INT, czyli liczba całkowita	Nie	Tak(order_user_fk)	Nie
IdCoupon	INT	przyjmowany m typem danych jest INT, czyli liczba całkowita	Nie	Tak(order_coupon_fk)	Nie



Fundusze Europejskie
Wiedza Edukacja Rozwój



**Rzeczpospolita
Polska**

Unia Europejska
Europejski Fundusz Społeczny



Zintegrowany Program Rozwoju Politechniki Lubelskiej – część druga

Idpaymentmethod	INT NOT NULL	kolumna nie może mieć wartości NULL, przyjmowany m typem danych jest INT, czyli liczba całkowita	Nie	Tak(order_paymentmethod_fk)	Nie
IdShop	INT NOT NULL	przyjmowany m typem danych jest INT, czyli liczba całkowita	Nie	Tak(order_shopinfo_fk)	Nie
IdCity	INT NOT NULL	przyjmowany m typem danych jest INT, czyli liczba całkowita	Nie	Tak(order_city_fk)	Nie
IdStatus	INT NOT NULL	przyjmowany m typem danych jest INT, czyli liczba całkowita	Nie	Tak(order_status_fk)	Nie

OrderItem					
Nazwa kolumny	Nazwa ograniczenia	Opis ograniczenia	PK	FK	Unikalność
IdOrderItem	INT NOT NULL	kolumna nie może mieć wartości NULL, przyjmowany m typem danych jest INT, czyli liczba całkowita	Tak(orderitem_pk)	Nie	Tak(orderitem_idorderunion)

Zintegrowany Program Rozwoju Politechniki Lubelskiej – część druga

Amount	INT NOT NULL	kolumna nie może mieć wartości NULL, przyjmowany m typem danych jest INT, czyli liczba całkowita	Nie	Nie	Nie
IdOrder	INT NOT NULL	kolumna nie może mieć wartości NULL, przyjmowany m typem danych jest INT, czyli liczba całkowita	Nie	Tak(orderitem_order_fk)	Nie
idpricebookentry	INT NOT NULL	kolumna nie może mieć wartości NULL, przyjmowany m typem danych jest INT, czyli liczba całkowita	Nie	Tak(orderitem_pricebookentry_fk)	Nie

PaymentMethod					
Nazwa kolumny	Nazwa ograniczenia	Opis ograniczenia	PK	FK	Unikalność
idpaymentmethod	INT NOT NULL	kolumna nie może mieć wartości NULL, przyjmowany m typem danych jest INT, czyli liczba całkowita	Tak(paymentmethod_pk)	Nie	Nie

Zintegrowany Program Rozwoju Politechniki Lubelskiej – część druga

paymentmet hodname	CLOB NOT NULL	przechowuje duże ilości plików typu character, musi zawierać wartość - nie może być puste	Nie	Tak	Nie
-----------------------	------------------	--	-----	-----	-----



**Fundusze
Europejskie**
Wiedza Edukacja Rozwój



**Rzeczpospolita
Polska**

Unia Europejska
Europejski Fundusz Społeczny



Pricebook					
Nazwa kolumny	Nazwa ograniczenia	Opis ograniczenia	PK	FK	Unikalność
idpricebook	INT NOT NULL	kolumna nie może mieć wartości NULL, przyjmowany m typem danych jest INT, czyli liczba całkowita	Tak(pricebookentry_pk)	Nie	Tak(idpricebookentry_un)
type	CLOB	przechowuje duże ilości plików typu character	Nie	Nie	Nie
startdate	DATE NOT NULL	przechowuje datę rozumianą jako rok, dzień, miesiąc, godzinę, minutę i sekundę, nie może być puse	Nie	Nie	Nie
enddate	DATE NOT NULL	przechowuje datę rozumianą jako rok, dzień, miesiąc, godzinę, minutę i sekundę, nie może być puse	Nie	Nie	Nie

Zintegrowany Program Rozwoju Politechniki Lubelskiej – część druga

description	CLOB	przechowuje duże ilości plików typu character	Nie	Nie	Nie
-------------	------	--	-----	-----	-----

Product					
Nazwa kolumny	Nazwa ograniczenia	Opis ograniczenia	PK	FK	Unikalność
idproduct	INT NOT NULL	kolumna nie może mieć wartości NULL, przyjmowany m typem danych jest INT, czyli liczba całkowita	Tak(prodcut_pk)	Nie	Tak(idproduct_un)
productname	CLOB	przechowuje duże ilości plików typu character	Nie	Nie	Nie
description	CLOB	przechowuje duże ilości plików typu character	Nie	Nie	Nie
Image	BLOB	przechowuje duże pliki binarne	Nie	Nie	Nie
amount	INT	przechowuje liczby całkowite	Nie	Nie	Nie



User					
Nazwa kolumny	Nazwa ograniczenia	Opis ograniczenia	PK	FK	Unikalność
iduser	VARCHAR NOT NULL	kolumna nie może mieć wartości NULL, przyjmowany m typem danych jest ciąg znaków o zmiennej długości	Tak(user_pk)	Nie	Nie
firstname	VARCHAR (50) NOT NULL	typem danych jest ciąg znaków o zmiennej długości, 50 bitów	Nie	Nie	Nie
lastname	VARCHAR (50) NOT NULL	przechowuje duże ilości plików typu character, 50 bitów	Nie	Nie	Nie
sex	VARCHAR (50) NOT NULL	typem danych jest ciąg znaków o zmiennej długości, 50 bitów	Nie	Nie	Nie
email	VARCHAR (50) NOT NULL	typem danych jest ciąg znaków o zmiennej długości, 50 bitów	Nie	Nie	Nie
phone	INT NOT NULL	kolumna nie może mieć wartości NULL, przyjmowany m typem danych jest INT, czyli liczba	Nie	Nie	Nie



Zintegrowany Program Rozwoju Politechniki Lubelskiej – część druga

		całkowita			
postcode	VARCHAR (50) NOT NULL	typem danych jest ciąg znaków o zmiennej długości, 50 bitów	Nie	Nie	Nie
street	VARCHAR (50) NOT NULL	typem danych jest ciąg znaków o zmiennej długości, 50 bitów	Nie	Nie	Nie
newsletter	CHAR(1) NOT NULL	przechowuje znaki o długości 1 bajta, nie może przyjmować wartości NULL	Nie	Nie	Nie
status	CHAR(1) NOT NULL	przechowuje znaki o długości 1 bajta, nie może przyjmować wartości NULL	Nie	Nie	Nie

ROZDZIAŁ 7. KOD SQL – TWORZENIE BAZY DANYCH

Przedstaw instrukcje do tworzenia poszczególnych tabel bazy danych.

Przedstaw instrukcje do tworzenia więzów integralności zarówno na poziomie poszczególnych kolumn jak i całych tabel.

Przedstaw instrukcje do tworzenia związków między tabelami.

Przedstaw instrukcje do tworzenia indeksów dla poszczególnych tabel.

Instrukcje do tworzenia poszczególnych tabel bazy danych:

```
CREATE TABLE category (  
    idcategory INTEGER NOT NULL,  
    name CLOB NOT NULL  
);
```

```
CREATE TABLE city (  
    idcity INTEGER NOT NULL,  
    cityname CLOB NOT NULL,  
    idcountry INTEGER NOT NULL  
);
```

```
CREATE TABLE country (  
    idcountry INTEGER NOT NULL,  
    countryname CLOB NOT NULL,  
    phone INTEGER  
);
```

```
CREATE TABLE coupon (  
    idcoupon INTEGER NOT NULL,  
    type CLOB,  
    value INTEGER NOT NULL,  
    code CLOB NOT NULL,  
    used CHAR(1) NOT NULL  
);
```

```
CREATE TABLE favouritelistitem (  
    idfavouritelistitem INTEGER NOT NULL,  
    pricebookentry INTEGER,  
    idproduct INTEGER,  
    iduser INTEGER NOT NULL  
);
```

```
CREATE TABLE "File" (  
    idfile INTEGER NOT NULL,  
    type CLOB,  
    path CLOB  
);
```



```
CREATE TABLE memberclub (  
    idmemberclub  NUMBER NOT NULL,  
    active        CHAR(1) NOT NULL,  
    name          CLOB NOT NULL,  
    loyaltypoints INTEGER NOT NULL  
);
```

```
CREATE TABLE "Order" (  
    idorder      INTEGER NOT NULL,  
    amount       INTEGER NOT NULL,  
    address      CLOB,  
    tax          INTEGER NOT NULL,  
    totalamount  NUMBER NOT NULL,  
    status       CHAR(1) NOT NULL,  
    refunds      CHAR(1),  
    postcode     INTEGER NOT NULL,  
    notification CHAR(1) NOT NULL,  
    iduser       INTEGER NOT NULL,  
    idcoupon     INTEGER,  
    idpaymentmethod INTEGER NOT NULL,  
    idcity       INTEGER NOT NULL,  
    idshop       INTEGER NOT NULL,  
    idstatus     INTEGER NOT NULL  
);
```

```
CREATE TABLE orderitem (  
    idorderitem  INTEGER NOT NULL,  
    amount       INTEGER NOT NULL,  
    idorder      INTEGER NOT NULL,  
    idpricebookentry INTEGER NOT NULL  
);
```

```
CREATE TABLE paymentmethod (  
    idpaymentmethod INTEGER NOT NULL,  
    paymentmethodname CLOB NOT NULL  
);
```

```
CREATE TABLE pricebook (  
    idpricebook INTEGER NOT NULL,  
    type        CLOB,  
    startdate   DATE NOT NULL,  
    enddate     DATE NOT NULL,
```



```
description CLOB  
);
```

```
CREATE TABLE pricebookentry (  
    idpricebookentry INTEGER NOT NULL,  
    pricebookprice FLOAT NOT NULL,  
    amount INTEGER,  
    idproduct INTEGER NOT NULL,  
    idpricebook INTEGER NOT NULL  
);
```

```
CREATE TABLE product (  
    idproduct INTEGER  
        CONSTRAINT nnc_product_idproduct NOT NULL,  
    code VARCHAR2(20)  
        CONSTRAINT nnc_product_code NOT NULL,  
    productname CLOB,  
    description CLOB,  
    image BLOB,  
    amount INTEGER  
        CONSTRAINT nnc_product_amount NOT NULL,  
    active CHAR(1)  
        CONSTRAINT nnc_product_active NOT NULL,  
    idtax INTEGER  
        CONSTRAINT nnc_product_idtax NOT NULL,  
    idfile INTEGER  
        CONSTRAINT nnc_product_idfile_2 NOT NULL,  
    idsubcategory INTEGER  
        CONSTRAINT nnc_product_dsubcategory NOT NULL,  
    idcategory INTEGER  
        CONSTRAINT nnc_product_idcategory NOT NULL  
);
```

```
CREATE TABLE refunds (  
    iduser INTEGER NOT NULL,  
    idrefunds NUMBER NOT NULL,  
    refunddate DATE NOT NULL,  
    idorder INTEGER NOT NULL,  
    idstatus INTEGER NOT NULL );
```

```
CREATE TABLE refundsitem (  
    idrefundsitem NUMBER NOT NULL,  
    idrefunds NUMBER NOT NULL,  
    idorderitem INTEGER NOT NULL,  
    amount INTEGER NOT NULL
```



);

```
CREATE TABLE role (  
    idrole INTEGER NOT NULL,  
    name CLOB  
    type CLOB NOT NULL,  
    iduser INTEGER NOT NULL
```

);

```
CREATE TABLE shopinfo (  
    idshop INTEGER NOT NULL,  
    shopname CLOB NOT NULL,  
    shoplocation CLOB NOT NULL  
);
```

```
CREATE TABLE shopsupply (  
    idsupply INTEGER NOT NULL,  
    idshop INTEGER NOT NULL,  
    idproduct INTEGER NOT NULL,  
    amount INTEGER  
);
```

```
CREATE TABLE status (  
    idstatus INTEGER NOT NULL,  
    type CLOB NOT NULL  
);
```

```
CREATE TABLE subcategory (  
    idsubcategory INTEGER NOT NULL,  
    name CLOB NOT NULL,  
    idcategory INTEGER NOT NULL  
);
```

```
CREATE TABLE tax (  
    idtax INTEGER NOT NULL,  
    value INTEGER NOT NULL  
);
```

```
CREATE TABLE "User" (  
    iduser    INTEGER NOT NULL,  
    firstname VARCHAR2(50) NOT NULL,  
    lastname  VARCHAR2(50) NOT NULL,  
    sex       VARCHAR2(50) NOT NULL,  
    email     VARCHAR2(100) NOT NULL,  
    phone     INTEGER NOT NULL,  
    postcode  INTEGER NOT NULL,  
    street    VARCHAR2(50) NOT NULL,  
    newsletter CHAR(1) NOT NULL,  
    status    CHAR(1) NOT NULL,  
    idmemberclub NUMBER NOT NULL,  
    idrole    INTEGER NOT NULL,  
    idcity    INTEGER NOT NULL  
);
```

Instrukcje do tworzenia więzów integralności zarówno na poziomie poszczególnych kolumn jak i całych tabel:

```
ALTER TABLE category ADD CONSTRAINT category_pk PRIMARY KEY ( idcategory );
```

```
ALTER TABLE category ADD CONSTRAINT idcategory_un UNIQUE ( idcategory );
```

```
ALTER TABLE city ADD CONSTRAINT city_pk PRIMARY KEY ( idcity );
```

```
ALTER TABLE country ADD CONSTRAINT country_pk PRIMARY KEY ( idcountry );
```

```
ALTER TABLE country ADD CONSTRAINT idcountry_un UNIQUE ( idcountry );
```

```
ALTER TABLE coupon ADD CONSTRAINT coupon_pk PRIMARY KEY ( idcoupon );
```

```
ALTER TABLE coupon ADD CONSTRAINT idcoupon_un UNIQUE ( idcoupon );
```

```
ALTER TABLE favouritelistitem ADD CONSTRAINT favouritelistitem_pk PRIMARY  
KEY ( idfavouritelistitem );
```

```
ALTER TABLE favouritelistitem ADD CONSTRAINT idfavouritelistitem_un UNIQUE ( idfavouritelistitem );
```

```
ALTER TABLE "File" ADD CONSTRAINT file_pk PRIMARY KEY ( idfile );
```

```
ALTER TABLE "File" ADD CONSTRAINT idfile_un UNIQUE ( idfile );
```

Zintegrowany Program Rozwoju Politechniki Lubelskiej – część druga

```
ALTER TABLE memberclub ADD CONSTRAINT memberclub_pk PRIMARY KEY (
idmemberclub );
```

```
ALTER TABLE memberclub ADD CONSTRAINT idmemberclub_un UNIQUE (
idmemberclub );
```

```
ALTER TABLE "Order" ADD CONSTRAINT order_pk PRIMARY KEY ( idorder );
```

```
ALTER TABLE "Order" ADD CONSTRAINT order_idorder_un UNIQUE ( idorder );
```

```
ALTER TABLE orderitem ADD CONSTRAINT orderitem_pk PRIMARY KEY (
idorderitem );
```

```
ALTER TABLE paymentmethod ADD CONSTRAINT paymentmethod_pk PRIMARY KEY
( idpaymentmethod );
```

```
ALTER TABLE pricebookentry ADD CONSTRAINT pricebookentry_pk PRIMARY KEY (
idpricebookentry );
```

```
ALTER TABLE pricebookentry ADD CONSTRAINT idpricebookentry_un UNIQUE (
idpricebookentry );
```

```
ALTER TABLE product ADD CONSTRAINT product_pk PRIMARY KEY ( idproduct );
```

```
ALTER TABLE product ADD CONSTRAINT idproduct_un UNIQUE ( idproduct );
```

```
ALTER TABLE refunds ADD CONSTRAINT refunds_pk PRIMARY KEY ( idrefunds );
```

```
ALTER TABLE refunds ADD CONSTRAINT idrefunds_un UNIQUE ( idrefunds );
```

```
ALTER TABLE refundsitem ADD CONSTRAINT refundsitem_pk PRIMARY KEY (
idrefundsitem );
```

```
ALTER TABLE refundsitem ADD CONSTRAINT idrefundsitem_un UNIQUE (
idrefundsitem );
```

```
ALTER TABLE role ADD CONSTRAINT role_pk PRIMARY KEY ( idrole );
```

```
ALTER TABLE shopinfo ADD CONSTRAINT shopinfo_pk PRIMARY KEY ( idshop );
```

```
ALTER TABLE shopsupply ADD CONSTRAINT shopsupply_pk PRIMARY KEY (
idsupply );
```

```
ALTER TABLE subcategory ADD CONSTRAINT subcategory_pk PRIMARY KEY (
idsubcategory,
```



```
ALTER TABLE subcategory ADD CONSTRAINT idsubcategory_un UNIQUE (
idsubcategory );
```

```
ALTER TABLE tax ADD CONSTRAINT tax_pk PRIMARY KEY ( idtax );
```

```
ALTER TABLE tax ADD CONSTRAINT idtax_un UNIQUE ( idtax );
```

```
ALTER TABLE "User" ADD CONSTRAINT user_pk PRIMARY KEY ( iduser );
```

Instrukcje do tworzenia związków między tabelami:

```
ALTER TABLE city
ADD CONSTRAINT city_country_fk FOREIGN KEY ( idcountry )
REFERENCES country ( idcountry );
```

```
ALTER TABLE favouritelistitem
ADD CONSTRAINT favouritelistitem_product_fk FOREIGN KEY ( idproduct )
REFERENCES product ( idproduct );
```

```
ALTER TABLE favouritelistitem
ADD CONSTRAINT favouritelistitem_user_fk FOREIGN KEY ( iduser )
REFERENCES "User" ( iduser );
```

```
ALTER TABLE "Order"
ADD CONSTRAINT order_city_fk FOREIGN KEY ( idcity )
REFERENCES city ( idcity );
```

```
ALTER TABLE "Order"
ADD CONSTRAINT order_coupon_fk FOREIGN KEY ( idcoupon )
REFERENCES coupon ( idcoupon );
```

```
ALTER TABLE "Order"
ADD CONSTRAINT order_paymentmethod_fk FOREIGN KEY ( idpaymentmethod )
REFERENCES paymentmethod ( idpaymentmethod );
```

```
ALTER TABLE "Order"
ADD CONSTRAINT order_shopinfo_fk FOREIGN KEY ( idshop )
REFERENCES shopinfo ( idshop );
```

```
ALTER TABLE "Order"
ADD CONSTRAINT order_status_fk FOREIGN KEY ( idstatus )
REFERENCES status ( idstatus );
```



ALTER TABLE "Order"

ADD CONSTRAINT order_user_fk FOREIGN KEY (iduser)
REFERENCES "User" (iduser);

ALTER TABLE orderitem

ADD CONSTRAINT orderitem_order_fk FOREIGN KEY (idorder)
REFERENCES "Order" (idorder);

ALTER TABLE orderitem

ADD CONSTRAINT orderitem_pricebookentry_fk FOREIGN KEY (idpricebookentry)
REFERENCES pricebookentry (idpricebookentry);

ALTER TABLE pricebookentry

ADD CONSTRAINT pricebookentry_pricebook_fk FOREIGN KEY (idpricebook)
REFERENCES pricebook (idpricebook);

ALTER TABLE pricebookentry

ADD CONSTRAINT pricebookentry_product_fk FOREIGN KEY (idproduct)
REFERENCES product (idproduct);

ALTER TABLE product

ADD CONSTRAINT product_file_fk FOREIGN KEY (idfile)
REFERENCES "File" (idfile);

ALTER TABLE product

ADD CONSTRAINT product_subcategory_fk FOREIGN KEY (idsubcategory,
idcategory)
REFERENCES subcategory (idsubcategory,
idcategory);

ALTER TABLE product

ADD CONSTRAINT product_tax_fk FOREIGN KEY (idtax)
REFERENCES tax (idtax);

ALTER TABLE refunds

ADD CONSTRAINT refunds_order_fk FOREIGN KEY (idorder)
REFERENCES "Order" (idorder);

ALTER TABLE refunds

ADD CONSTRAINT refunds_status_fk FOREIGN KEY (idstatus)
REFERENCES status (idstatus);

ALTER TABLE refunds

ADD CONSTRAINT refunds_user_fk FOREIGN KEY (iduser)
REFERENCES "User" (iduser);



```
ALTER TABLE refundsitem  
  ADD CONSTRAINT refundsitem_orderitem_fk FOREIGN KEY ( idorderitem )  
  REFERENCES orderitem ( idorderitem );
```

```
ALTER TABLE refundsitem  
  ADD CONSTRAINT refundsitem_refunds_fk FOREIGN KEY ( idrefunds )  
  REFERENCES refunds ( idrefunds );
```

```
ALTER TABLE shopsupply  
  ADD CONSTRAINT shopsupply_product_fk FOREIGN KEY ( idproduct )  
  REFERENCES product ( idproduct );
```

```
ALTER TABLE shopsupply  
  ADD CONSTRAINT shopsupply_shopinfo_fk FOREIGN KEY ( idshop )  
  REFERENCES shopinfo ( idshop );
```

```
ALTER TABLE subcategory  
  ADD CONSTRAINT subcategory_category_fk FOREIGN KEY ( idcategory )  
  REFERENCES category ( idcategory );
```

```
ALTER TABLE "User"  
  ADD CONSTRAINT user_city_fk FOREIGN KEY ( idcity )  
  REFERENCES city ( idcity );
```

```
ALTER TABLE "User"  
  ADD CONSTRAINT user_memberclub_fk FOREIGN KEY ( idmemberclub )  
  REFERENCES memberclub ( idmemberclub );
```

```
ALTER TABLE "User"  
  ADD CONSTRAINT user_role_fk FOREIGN KEY ( idrole )  
  REFERENCES role ( idrole );
```

Instrukcje do tworzenia indeksów dla poszczególnych tabel:

```
CREATE UNIQUE INDEX product_idfile_2_idx ON product ( idfile ASC );
```

```
CREATE UNIQUE INDEX refundsitem__idx ON refundsitem (idorderitem ASC );
```

```
CREATE UNIQUE INDEX user__idx ON "User" (idmemberclub ASC );
```

Instrukcje do dodawania danych do tabel:

INSERT INTO Role ('idrole', 'name', 'type', 'iduser') VALUES (3, 'moderator', 'pracownik', 3);

INSERT INTO Role ('idrole', 'name', 'type', 'iduser') VALUES (5, 'klient', 'osobaZewnetrzna', 1);

INSERT INTO Role ('idrole', 'name', 'type', 'iduser') VALUES (2, 'sprzedawca', 'pracownik', 2);

INSERT INTO Role ('idrole', 'name', 'type', 'iduser',) VALUES (1, 'administrator', 'pracownik', 4);

INSERT INTO User ('User ID', 'FirstName', 'LastName', 'Sex', 'Email', 'Phone', 'PostCode', 'Street', 'Newsletter', 'Status', 'idmemberclub', 'idrole', 'idcity') VALUES (1, 'Jan', 'Kowalski', 'M', 'jan.kowalski@gmail.com', '0048515876908', '22200', 'Zeromskiego 45', '0', '1', 112001, 5, 345);

INSERT INTO User ('User ID', 'FirstName', 'LastName', 'Sex', 'Email', 'Phone', 'PostCode', 'Street', 'Newsletter', 'Status', 'idmemberclub', 'idrole', 'idcity') VALUES (2, 'Kuba', 'Nowakowski', 'M', 'kuba.nowakowski@gmail.com', '0048675923678', '21040', 'Jana Pawła 45', '1', '2', 000000, 2, 343);

INSERT INTO User ('User ID', 'FirstName', 'LastName', 'Sex', 'Email', 'Phone', 'PostCode', 'Street', 'Newsletter', 'Status', 'idmemberclub', 'idrole', 'idcity') VALUES (3, 'Justyna', 'Paradowska', 'K', 'justyna.paradowska@gmail.com', '0048765128463', '32040', 'Andersa 48', '1', '0', 321567, 3, 963);

INSERT INTO User ('User ID', 'FirstName', 'LastName', 'Sex', 'Email', 'Phone', 'PostCode', 'Street', 'Newsletter', 'Status', 'idmemberclub', 'idrole', 'idcity') VALUES (4, 'Malgorzata', 'Kula', 'K', 'malgorzata.kula@gmail.com', '0048890765241', '55015', 'Polna 24', '0', '1', 000000, 1, 321);

INSERT INTO Product ('idproduct', 'code', 'productname', 'description', 'image', 'amount', 'active', 'idtax', 'idfile', 'subcategory', 'idcategory') VALUES (1, 5468KI93, 'iPhone8', 'The best of this year', 'iphone8.jpg', 100, 1, 310,11,4,10);

INSERT INTO Product ('idproduct', 'code', 'productname', 'description', 'image', 'amount', 'active', 'idtax', 'idfile', 'subcategory', 'idcategory') VALUES (2, 2428LE91, 'iPhonePro', 'The newest of this year', 'iphonepro.jpg', 100, 1, 310,12,4,10);

INSERT INTO Product ('idproduct', 'code', 'productname', 'description', 'image', 'amount', 'active', 'idtax', 'idfile', 'subcategory', 'idcategory') VALUES (3, PE1743F, 'iPadMini', 'The best of this year', 'ipadmini.jpg', 500, 0, 310,15,6,11);

INSERT INTO Product ('idproduct', 'code', 'productname', 'description', 'image', 'amount', 'active', 'idtax', 'idfile', 'subcategory', 'idcategory') VALUES (4, 2315OR93, 'iPadPro', 'Impossible achievement', 'ipadpro.jpg', 500, 1, 310,16,7,11);



INSERT INTO Order('idorder', 'amount', 'address', 'tax', 'totalamount', 'status', 'refunds', 'postcode', 'notification', 'iduser', 'idcoupon', 'idpaymentmethod', 'idcity', 'idshop', 'idstatus') VALUES (250, 1, "Wojska Polskiego 44/12", 310, 1874.09, 1, 0, 4, 764234, 2, 321, 1, 5);

INSERT INTO Order('idorder', 'amount', 'address', 'tax', 'totalamount', 'status', 'refunds', 'postcode', 'notification', 'iduser', 'idcoupon', 'idpaymentmethod', 'idcity', 'idshop', 'idstatus') VALUES (125, 2, "Chrobrego 5/25", 310, 4562.77, 0, 0,);

INSERT INTO Country('idcountry', 'countryname', 'phone') VALUES (241, 'Poland', 48)

INSERT INTO Country('idcountry', 'countryname', 'phone') VALUES (230, 'Germany', 49);





Raport powstał podczas zajęć laboratoryjnych z przedmiotu
prowadzonego w ramach projektu
*„Zintegrowany Program Rozwoju Politechniki Lubelskiej – część
druga”*,
umowa nr **POWR.03.05.00-00-Z060/18-00**
w ramach Programu Operacyjnego Wiedza Edukacja Rozwój
2014-2020
współfinansowanego ze środków Europejskiego Funduszu
Społecznego