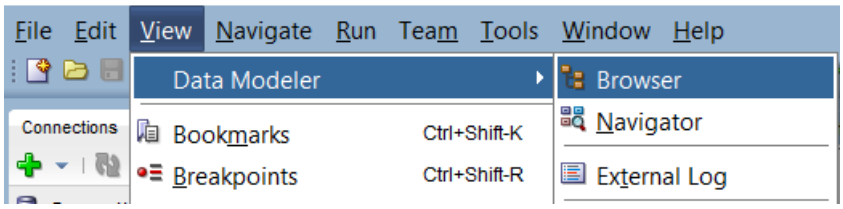
Zestaw 9
Modelowanie bazy danych

Przy wykonaniu poniższych ćwiczeń ***nie korzystamy*** *z tabel utworzonych za pomocą skryptu SUMMIT.SQL. Inaczej też wykorzystujemy narzędzie, jakim jest SQL Developer – nie tylko jako konsolę do wprowadzania poleceń języka SQL, ale przede wszystkim jako program do tworzenia modeli baz danych (*Data Modeler*).*

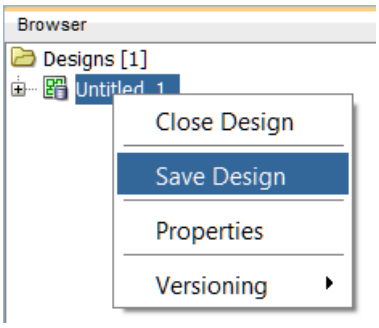
Przydatna literatura:
Z. Łojewski, Bazy danych – teoria i praktyka*, rozdział 7 (model związków encji);*
Ch. Murray, Oracle SQL Developer Data Modeler User's Guide, Release 4.1 *(**główny podręcznik**).*

Przygotowanie środowiska

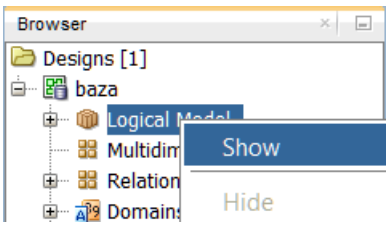
Włączyć w programie *SQL Developer* widok przeglądarki projektów *Data Modeler*, aktywując okno **Browser** za pomocą sekwencji poleceń z menu: **View → Data Modeler → Browser**.



Wykorzystać projekt *Untitled_1* jako kanwę nowego projektu, który można zapisać na dysku ze zmianą nazwy (np. na *baza*). W tym celu kliknąć prawym przyciskiem myszki na nazwie, a następnie wybrać *Save Design*.

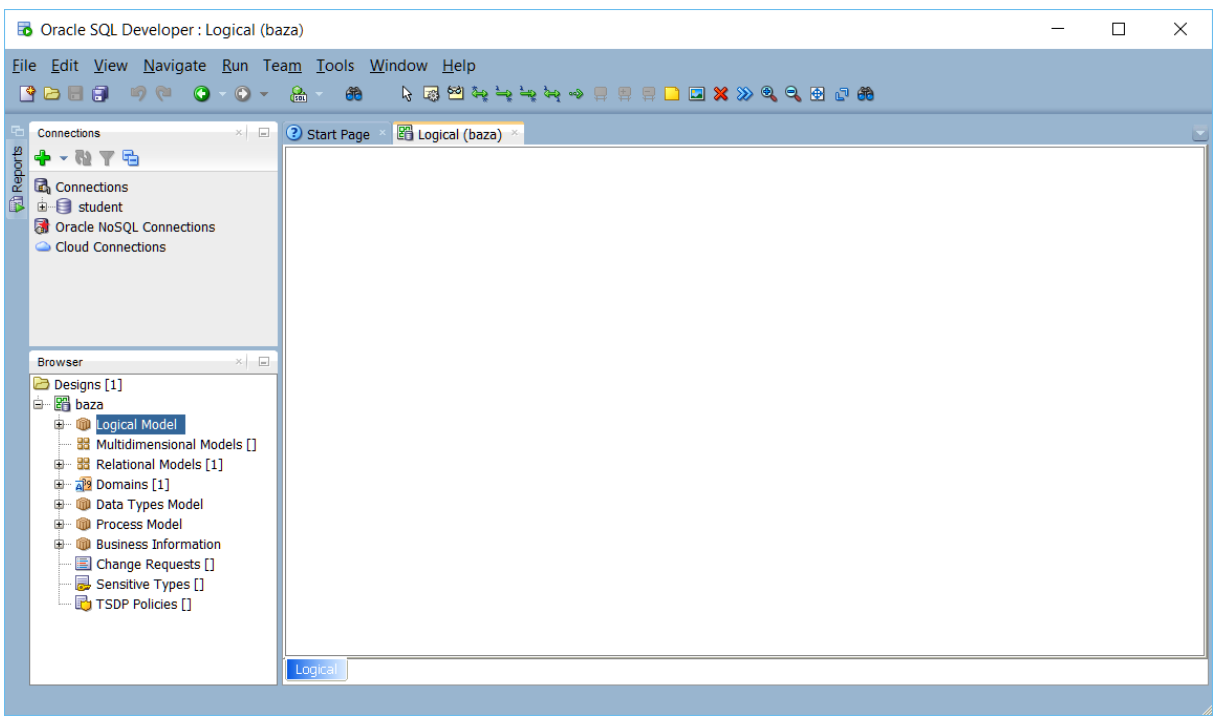


W pojawiającym się oknie dialogowym wybrać odpowiedni katalog i określić własną nazwę projektu.



Kolejnym krokiem jest rozpoczęcie pracy z modelem logicznym. Po rozwinięciu elementów składowych projektu znakiem + przy jego nazwie uzyskamy dostęp do pozycji *Logical Model*. Prawy klawisz myszki na nazwie umożliwia wybranie opcji *Show* z kontekstowego menu.

Spowoduje to otwarcie nowej zakładki *Logical (baza)* w oknie głównym programu:

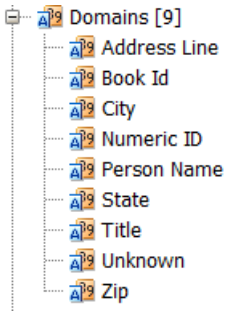


Tworzenie modelu logicznego

Model logiczny przykładowej bazy danych tworzony jest zgodnie z opisem zawartym w podrozdziale [2.1] podręcznika *Oracle SQL Developer Data Modeler User's Guide*.

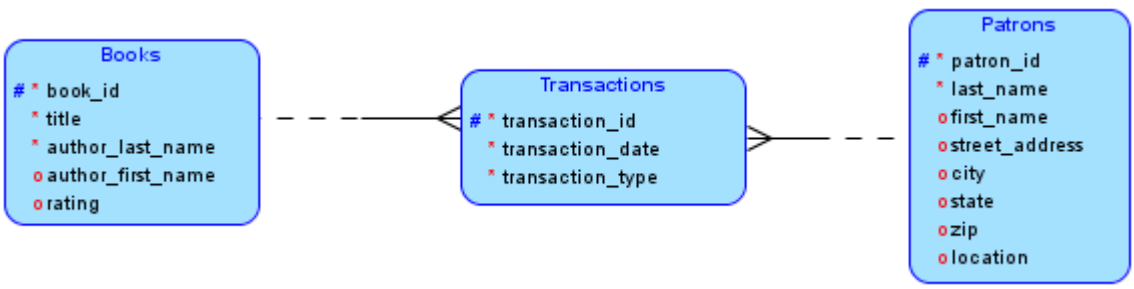
Należy wykonać następujące czynności:

- dodać domeny, które będą wykorzystywane przy definiowaniu encji (wypełniając tylko wskazane w opisie pola, a pomijając pozostałe); [2.1.1]



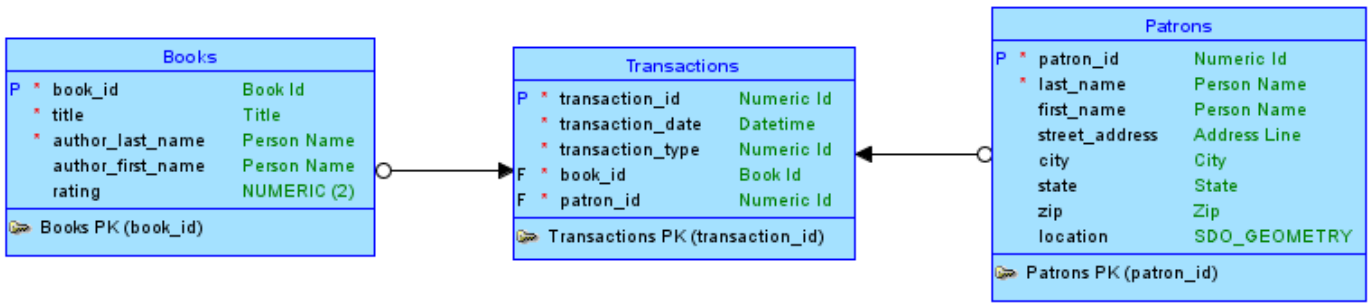
- utworzyć encje książek (*Books*), klientów biblioteki (*Patrons*) oraz rejestru wypożyczeń (*Transactions*); [2.1.2-2.1.4]

- utworzyć związki między encjami; [2.1.5]



Powyższy diagram przedstawiony jest w notacji Barkera.

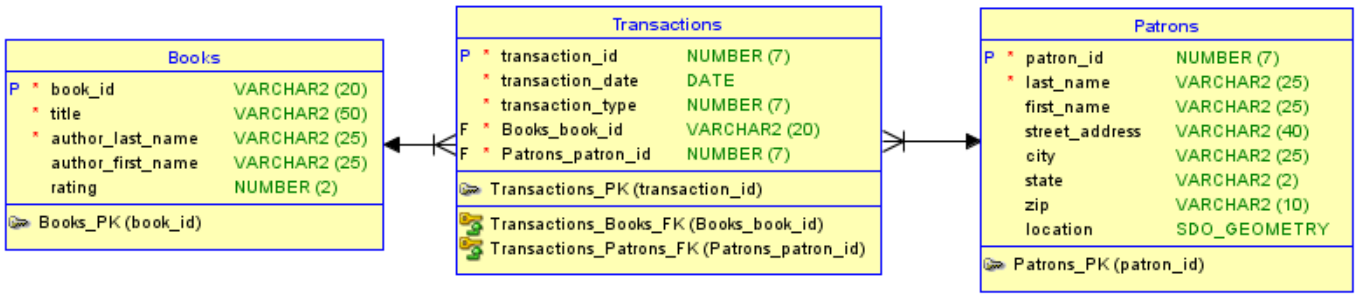
Więcej informacji zawiera schemat w notacji Bachmana, którą można uzyskać poprzez kontekstowe menu (po kliknięciu prawym przyciskiem myszki w obszarze edycyjnym modelu logicznego) i wybór opcji **Notation → Bachman Notation**.



Tworzenie modelu relacyjnego

Zgodnie z opisem w rozdziale [2.2] podręcznika Oracle, przejście do modelu relacyjnego jest łatwo realizowane poprzez wskazanie modelu logicznego i wybranie funkcji **Engineer to Relational Model**.

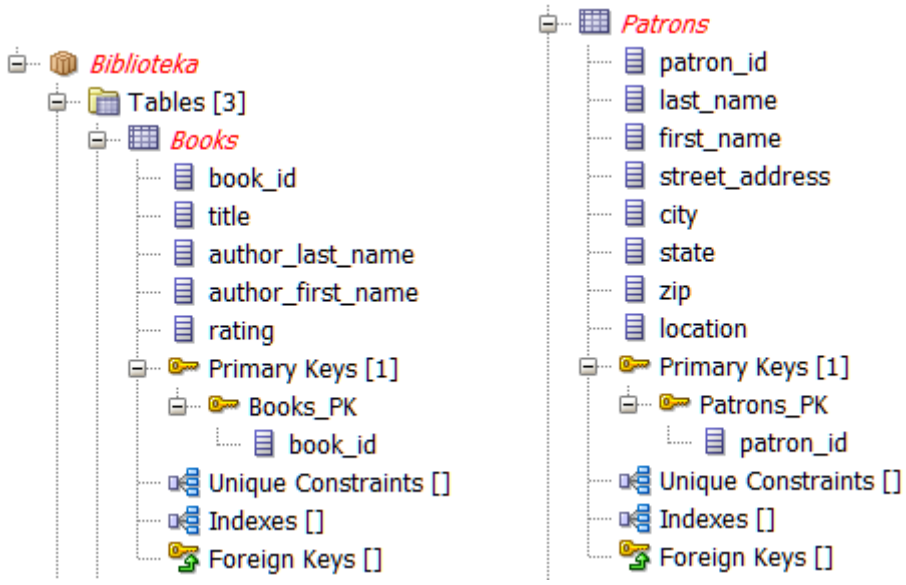
Nie stosujemy w tym przypadku filtrowania (tzn. wybieramy wszystkie składowe).

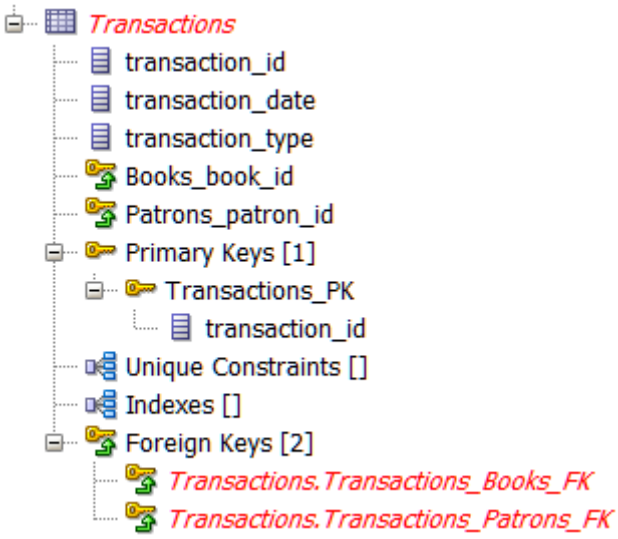


Model fizyczny

Przed utworzeniem kodu SQL warto zobaczyć fizyczny model projektu. Zgodnie z zapisami w rozdziale [2.3] powinniśmy wybrać odpowiedni typ bazy danych – w tym przypadku Oracle Database 11g.

Po utworzeniu modelu fizycznego możemy obejrzeć składowe, w tym tabele, poprzez rozwinięcie odpowiednich pozycji w przeglądarce.





Uzyskiwanie kodu SQL

Proces tworzenia kodu opisany jest w rozdziale [2.3] podręcznika. Należy wygenerować instrukcje DDL (*Data Definition Language*), zapisać do pliku, a następnie można je wykorzystać jako skrypt do tworzenia tabel bazy danych.

Przykładowy kod SQL:

```
-- Generated by Oracle SQL Developer Data Modeler 4.1.3.901
--   at:        2016-05-12 01:57:22 CEST
--   site:      Oracle Database 11g
--   type:      Oracle Database 11g

CREATE TABLE Books
  (
    book_id           VARCHAR2 (20) NOT NULL ,
    title             VARCHAR2 (50) NOT NULL ,
    author_last_name  VARCHAR2 (25) NOT NULL ,
    author_first_name VARCHAR2 (25) ,
    rating            NUMBER (2)
  )
  LOGGING ;
ALTER TABLE Books ADD CONSTRAINT Books_PK PRIMARY KEY ( book_id ) ;


CREATE TABLE Patrons
  (
    patron_id      NUMBER (7) NOT NULL ,
    last_name      VARCHAR2 (25) NOT NULL ,
    first_name     VARCHAR2 (25) ,
    street_address VARCHAR2 (40) ,
    city           VARCHAR2 (25) ,
    state          VARCHAR2 (2) ,
    zip            VARCHAR2 (10) ,
    location MDSYS.SDO_GEOMETRY
  )
  LOGGING ;
ALTER TABLE Patrons ADD CONSTRAINT Patrons_PK PRIMARY KEY ( patron_id ) ;

CREATE TABLE Transactions
  (
    transaction_id    NUMBER (7) NOT NULL ,
    transaction_date  DATE NOT NULL ,
    transaction_type  NUMBER (7) NOT NULL ,
    Books_book_id     VARCHAR2 (20) NOT NULL ,
    Patrons_patron_id NUMBER (7) NOT NULL
  )
  LOGGING ;
ALTER TABLE Transactions ADD CONSTRAINT Transactions_PK PRIMARY KEY ( transaction_id ) ;

ALTER TABLE Transactions ADD CONSTRAINT Transactions_Books_FK FOREIGN KEY ( Books_book_id ) REFERENCES
Books ( book_id ) NOT DEFERRABLE ;

ALTER TABLE Transactions ADD CONSTRAINT Transactions_Patrons_FK FOREIGN KEY ( Patrons_patron_id )
REFERENCES Patrons ( patron_id ) NOT DEFERRABLE ;
```

Tak utworzony projekt warto zapisać zgodnie z instrukcjami w podręczniku.

Uwaga: przy pracy z rozszerzeniem Data Modeler w programie SQL Developer należy posługiwać się opcją zapisu **File → Data Modeler → Save** *(a nie* **File → Save**, która zazwyczaj w takim przypadku nie jest aktywna).

Zadanie

Opracować model własnej bazy danych, realizując kolejne kroki, które umożliwiają podgląd modelu (logicznego, relacyjnego i fizycznego) oraz uzyskanie kodu SQL. Baza powinna zawierać kilka encji (tabel), powiązanych ze sobą w sensowny sposób.

Jako rozwiązanie przesłać plik tekstowy (PDF) z opisem, graficzną reprezentacją poszczególnych modeli oraz kodem SQL.