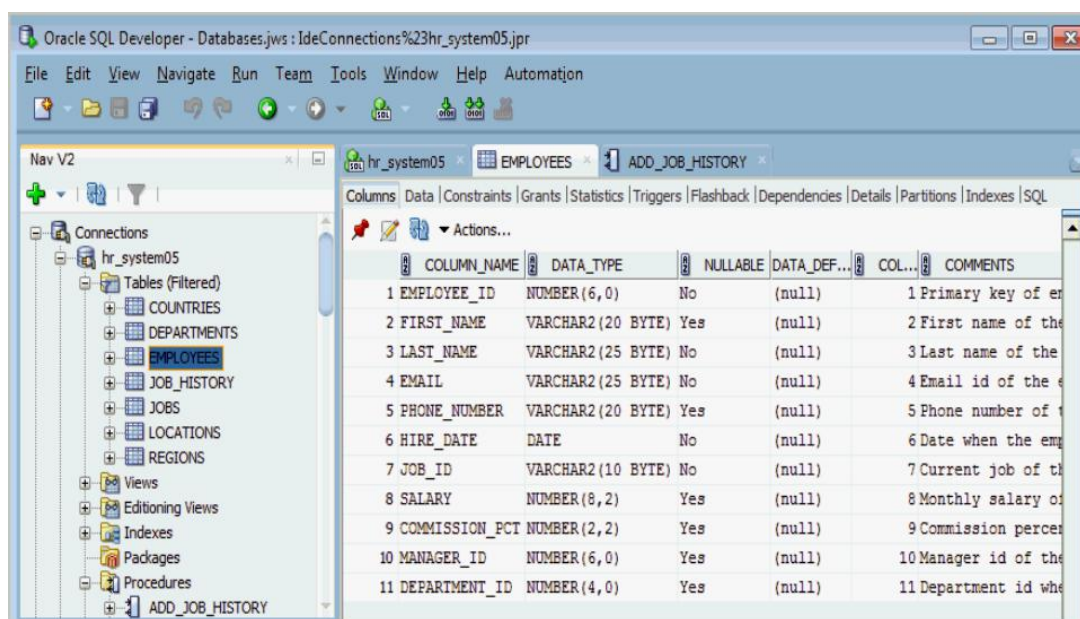


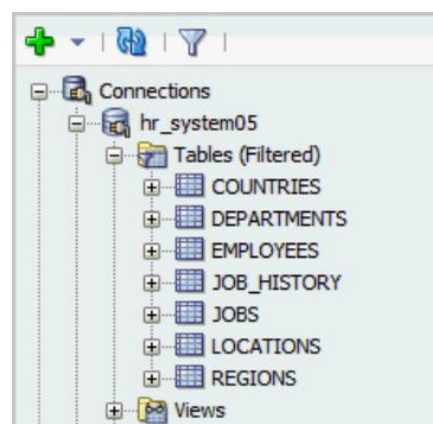
SQL Developer

SQL Developer jest narzędziem pozwalającym na wykonywanie podstawowych operacji związanych z dostępem do bazy danych np. przeglądanie danych, tworzenie tabel i obiektów bazy, modyfikowanie tych obiektów, monitorowanie działania bazy danych, tworzenie procedur / funkcji składowanych.

W lewej części panelu SQL Developera znajduje się okno do nawigacji i wyszukiwania obiektów bazy danych, w prawej części panelu znajdują się okna pokazujące stan zaznaczonych obiektów. Na diagramie pokazano główny panel programu.



Lewa część panelu użytkownika zawiera opcje manipulowania połączeniami (**Connections**), folderami (**Files**) oraz raportami (**Reports**). Rys. 2 przedstawia okno połączeń.



Przebieg ćwiczenia

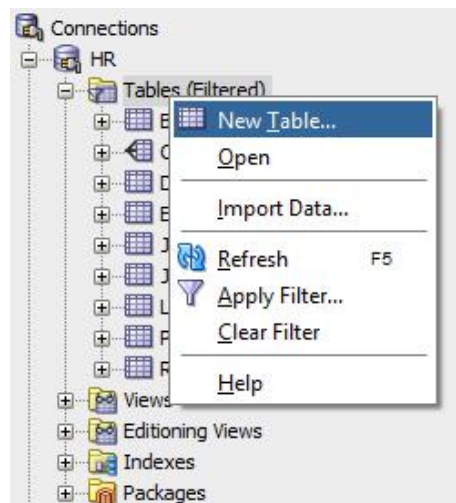
Przebieg ćwiczenia oparty jest o *tutorial* zawarty w dokumentacji *Oracle SQL Developer User's Guide*.

Tworzenie tabeli BOOKS

Klikamy prawym przyciskiem myszy na *Tables* w menu połączenia i wybieramy *New Table*. Następnie wprowadzamy poniższe informacje:

Warto zaznaczyć, że **klucz główny** jest taką wartością, która pozwala w sposób jednoznaczny zidentyfikować każdy rekord (wiersz) w tabeli. Z tego względu wartości klucza głównego nie mogą się powtarzać (muszą być unikalne).

Name: Books



| Nazwa kolumny | Typ danych | Rozmiar | Dodatkowe informacje |
|-------------------|------------|---------|---|
| Book_id | VARCHAR2 | 20 | Klucz główny tabeli (Automatycznie zostanie nałożone ograniczenie Not null, Unique oraz zostanie stworzony indeks dla tej kolumny). |
| title | VARCHAR2 | 50 | Ograniczenie Not null |
| Author_last_name | VARCHAR2 | 30 | Ograniczenie Not null |
| Author_first_name | VARCHAR2 | 30 | |
| Rating | NUMBER | | Subiektywna ocena książki (w skali od 1 do 10). |

Schema: HR

Name: BOOKS

Table: DDL

Columns:

| PK | Name | Data Type | Size | Not Null | Default | Comment |
|----|-------------------|-----------|------|-------------------------------------|---------|---------------|
| | BOOK_ID | VARCHAR2 | 20 | <input checked="" type="checkbox"/> | | Klucz główny |
| | TITLE | VARCHAR2 | 50 | <input checked="" type="checkbox"/> | | Tytuł książki |
| | AUTHOR_LAST_NAME | VARCHAR2 | 30 | <input checked="" type="checkbox"/> | | Dane autora |
| | AUTHOR_FIRST_NAME | VARCHAR2 | 30 | <input checked="" type="checkbox"/> | | Dane autora |
| | RATING | NUMBER | | <input type="checkbox"/> | | Ocena książki |

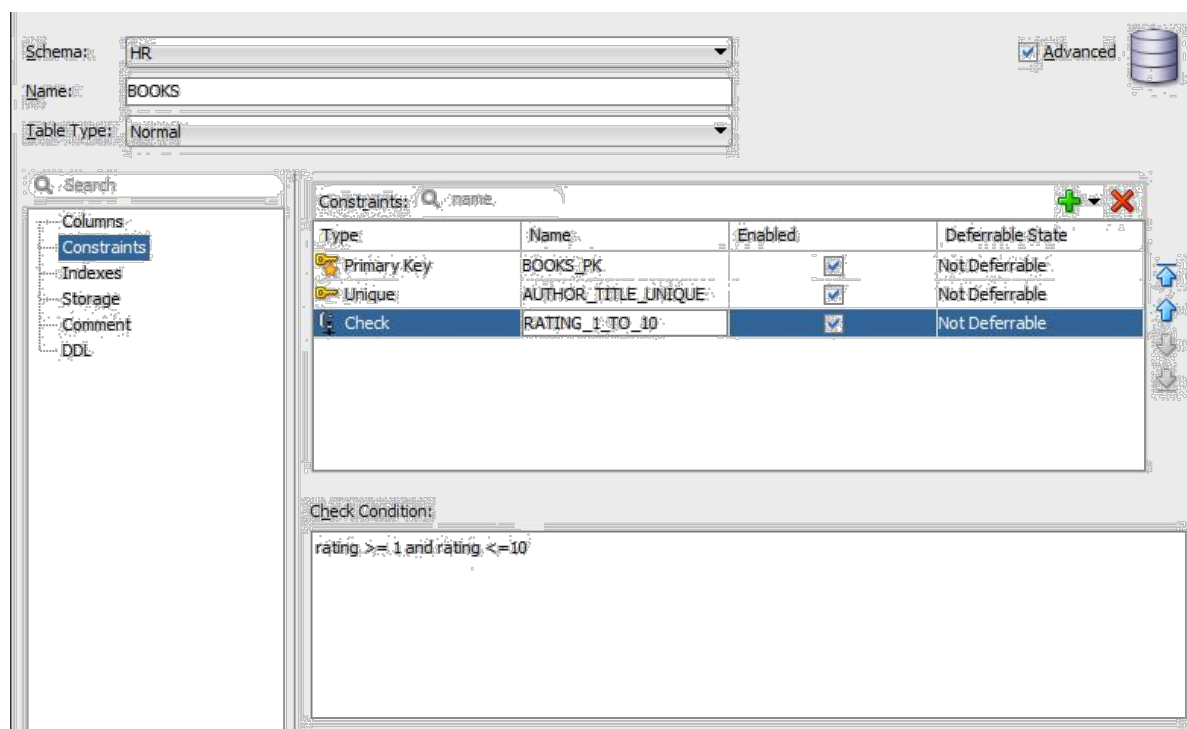
Po wprowadzeniu ostatniej kolumny (**rating**) zaznaczamy pole *Advanced* (obok menu *Schema*). Zostanie wyświetlone więcej opcji tworzenia tabeli. Dla naszej tabeli będzie potrzebne dodanie ograniczenia *Unique Constraint* (które zapewnia niepowtarzalność wartości w danej kolumnie) oraz *Check Constraint* (które pozwala zdefiniować predykat, który musi być spełniony podczas wprowadzania danych dla danej kolumny).

Dodawanie ograniczenia Unique Constraint

Przechodzimy do panelu *Constraints* i dodajemy nowe ograniczenie typu *New Unique Constraint*. Wprowadzamy nazwę ograniczenia `AUTHOR_TITLE_UNIQUE`, i wybieramy dla niego kolumny `TITLE` oraz `AUTHOR_LAST_NAME`. Tworzymy ograniczenie. System automatycznie zdefiniuje indeks dla tego ograniczenia.

Dodawanie ograniczenia Check Constraint

Ponownie dodajemy ograniczenie, ale tym razem typu *New Check Constraint*. Wprowadzamy nazwę ograniczenia `RATING_1_TO_10` i określamy predykat (w polu *Check Condition*) jako `rating >= 1 and rating <= 10`.



Dla porównania: polecenie SQL tworzące tabelę **BOOKS**.

CREATE TABLE books

```
(
  book_id      VARCHAR2(20),
  title        VARCHAR2(50) CONSTRAINT title_not_null NOT NULL,
  author_last_name VARCHAR2(30) CONSTRAINT last_name_not_null NOT NULL,
  author_first_name VARCHAR2(30),
  rating       NUMBER,
  CONSTRAINT books_pk PRIMARY KEY (book_id),
  CONSTRAINT rating_1_to_10 CHECK (rating IS NULL
    OR (rating          >= 1
    AND rating          <= 10)),
  CONSTRAINT author_title_unique UNIQUE (author_last_name, title)
);
```

Tworzenie tabeli PATRONS

Tabela **PATRONS** zawiera wpis dla każdej osoby mogącej wypożyczać książki. Definicja tabeli zawiera kilka typów prostych. Ponownie klikamy prawym przyciskiem myszy na menu *Tables* i wybieramy *Create New Table*. Definiujemy kolumny według poniższej tabeli:

| Nazwa kolumny | Typ | Rozmiar | Dodatkowe informacje |
|----------------|----------|---------|----------------------|
| Patron_id | NUMBER | | Klucz główny |
| Last_name | VARCHAR2 | 30 | Not null |
| First_name | VARCHAR2 | 30 | |
| Street_address | VARCHAR2 | 30 | |
| City_state_zip | VARCHAR2 | 30 | |
| Location | VARCHAR2 | 100 | |

Dla porównania poniżej przedstawiono polecenie SQL tworzące tabelę **PATRONS**.

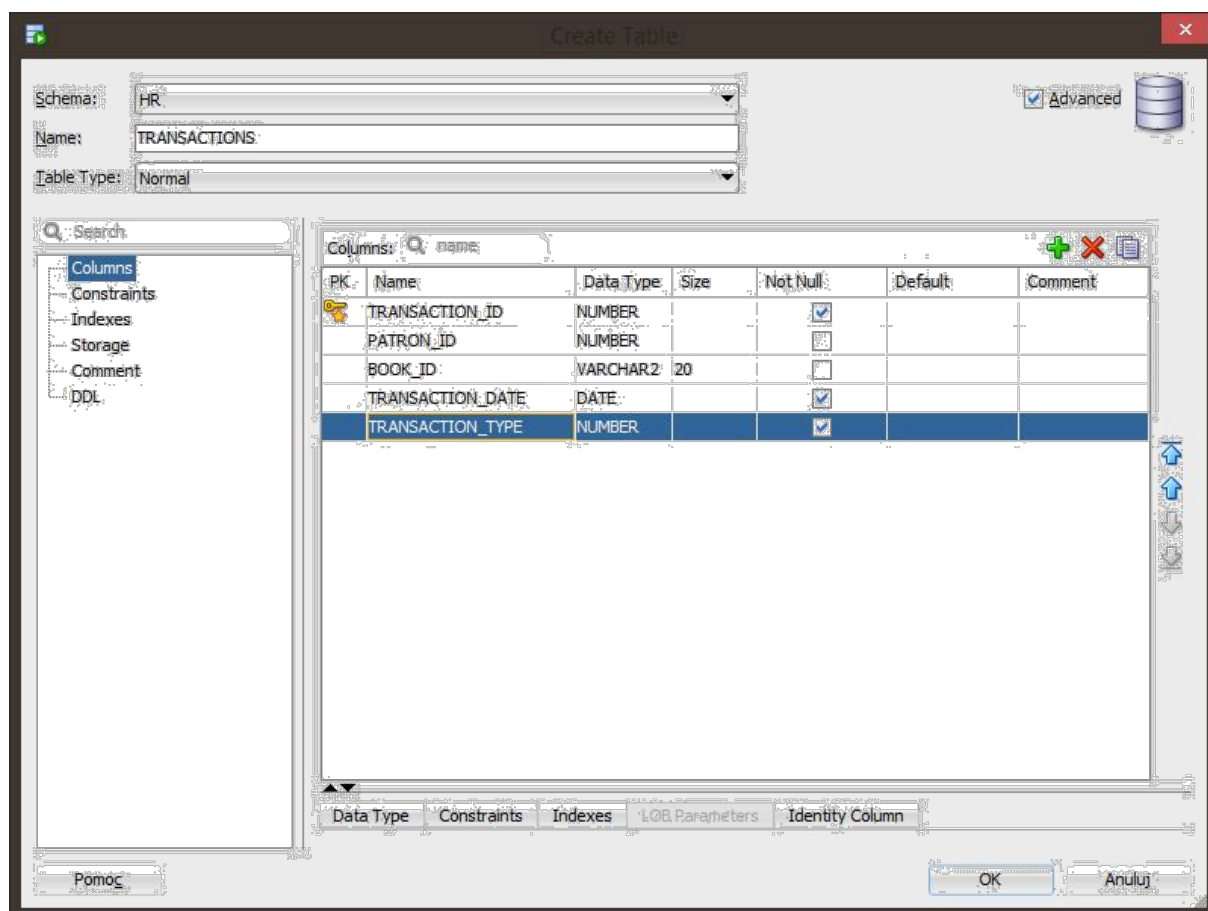
```
CREATE TABLE patrons
(
  patron_id    NUMBER,
  last_name    VARCHAR2(30) CONSTRAINT patron_last_not_null NOT NULL,
  first_name   VARCHAR2(30),
  street_address VARCHAR2(50),
  city_state_zip VARCHAR2(50),
  location     VARCHAR2(100),
  CONSTRAINT patrons_pk PRIMARY KEY (patron_id)
);
```

Tworzenie tabeli TRANSACTIONS

Tabela **Transactions** będzie zawierać po jednym wierszu dla każdego wypożyczenia lub zwrotu książki. Istotna jest tutaj informacja o wypożyczającym (tabela **Patrons**) oraz wypożyczanej książce (tabela **Books**). Dlatego tabela **Transactions** będzie zawierać tzw. **klucze obce** odnoszące się do tabel **Patrons** oraz **Books**. Ponownie klikamy prawym przyciskiem myszy na menu *Tables* i wybieramy *New Table*. Definiujemy kolumny jak poniżej:

| Nazwa kolumny | Typ | Rozmiar | Dodatkowe informacje |
|-------------------------|----------|---------|--|
| Transaction_id | NUMBER | | Klucz główny |
| Patron_id | NUMBER | | Klucz obcy (wartości w tej kolumnie muszą odpowiadać wartościom kolumny patron_id w tabeli Patrons) |
| Book_id | VARCHAR2 | 20 | Klucz obcy (wartości w tej kolumnie muszą odpowiadać wartościom kolumny book_id w tabeli Books) |
| Transaction_date | DATE | | Data i czas transakcji; not null |
| Transaction_type | NUMBER | | Cyfra określająca typ transakcji (np. 1 oznacza wypożyczenie); not null |

Rysunek poniżej przedstawia zdefiniowane kolumny (nie wciskamy jeszcze **OK!**).



Ponadto chcielibyśmy, żeby wartości kolumny TRANSACTION_ID były generowane automatycznie podczas wprowadzania danych do tej tabeli. Można w tym celu wykorzystać **Sekwencje**.

Sekwencja to obiekt, który pozwala na automatyczne generowanie wartości, które mogą zostać wykorzystane np. do wypełniania pewnych kolumn tabeli. Podczas definicji sekwencji określa się jej wartość początkową, wartość końcową oraz krok. Zazwyczaj krok wynosi po prostu 1, co oznacza, że każda kolejna wartość „pobrana” z sekwencji będzie większa o 1 od poprzedniej. Istnieje także możliwość określenia sekwencji jako cyklicznej, co oznacza, że kolejną wartością po „pobraniu” wartości końcowej będzie wartość początkowa.

W celu określenia sekwencji dla kolumny należy zaznaczyć z paska na dole ekranu należy wybrać *Identity Column*, a następnie w menu *Type : Column Sequence*. W polu *trigger* pojawi się domyślnie utworzona nazwa **Wyzwalacza**.

Wyzwalacz jest to obiekt bazy danych, który uruchamiany jest (wyzwalany) po wystąpieniu pewnych zdarzeń związanych np. z modyfikacją danych, albo utworzeniem nowej tabeli. Jednym z przykładów zastosowania wyzwalacza jest automatyczne wprowadzanie danych do pewnych kolumn tabeli. Taką daną może być np. wartość pobrana z sekwencji.

Ostatecznie określenie kolumny TRANSACTION_ID jako *Column Sequence* przedstawia rysunek poniżej.

Schema: HR
Name: TRANSACTIONS
Table Type: Normal

Columns:

| PK | Name | Data Type | Size | Not Null | Default | Comment |
|-------------------------------------|------------------|-----------|------|-------------------------------------|---------|---------|
| <input checked="" type="checkbox"/> | TRANSACTION_ID | NUMBER | | <input checked="" type="checkbox"/> | | |
| <input type="checkbox"/> | PATRON_ID | NUMBER | | <input type="checkbox"/> | | |
| <input type="checkbox"/> | BOOK_ID | VARCHAR2 | 20 | <input type="checkbox"/> | | |
| <input type="checkbox"/> | TRANSACTION_DATE | DATE | | <input checked="" type="checkbox"/> | | |
| <input type="checkbox"/> | TRANSACTION_TYPE | NUMBER | | <input checked="" type="checkbox"/> | | |

Type: Column Sequence
Trigger: TRANSACTIONS_TRG
Sequence Schema: HR

OK Anuluj

System automatycznie wygeneruje wyzwalacz o nazwie TRANSACTIONS_TRG.

Do utworzenia tabeli pozostało jeszcze zdefiniowanie wcześniej wspomnianych kluczy obcych. Klucze obce stanowią ograniczenie nakładane na tabelę, dlatego należy przejść do panelu *Constraints*.

Definiowanie klucza obcego dla kolumny PATRON_ID

Wybieramy nowe ograniczenie typu *New Foreign Key Constraint*, któremu nadajemy nazwę `for_key_patron_id`. Pozostałe parametry ograniczenia ustawiamy jak na rysunku poniżej.

Schema: Name: Table Type:

Search:

Columns: Constraints Indexes Storage Comment DDL

Constraints:

| Type | Name | Enabled | Deferrable State |
|-------------|-------------------|-------------------------------------|------------------|
| Primary Key | TRANSACTIONS_PK | <input checked="" type="checkbox"/> | Not Deferrable |
| Foreign Key | FOR_KEY_PATRON_ID | <input checked="" type="checkbox"/> | Not Deferrable |

Referenced Constraint:

Schema: Table: Constraint: On Delete:

Associations:

| Local Column | Referenced Column |
|--------------|-------------------|
| PATRON_ID | PATRON_ID |

Pomoc OK Anuluj

Definiowanie klucza obcego dla BOOK_ID

Postępujemy analogicznie jak wyżej, przy czym ograniczenie nazywamy `for_key_book_id`, a parametry ograniczenia ustawiamy jak na rysunku poniżej.

Schema: Name: Table Type:

Search:

Columns: Constraints Indexes Storage Comment DDL

Constraints:

| Type | Name | Enabled | Deferrable State |
|-------------|-------------------|-------------------------------------|------------------|
| Primary Key | TRANSACTIONS_PK | <input checked="" type="checkbox"/> | Not Deferrable |
| Foreign Key | FOR_KEY_PATRON_ID | <input checked="" type="checkbox"/> | Not Deferrable |
| Foreign Key | FOR_KEY_BOOK_ID | <input checked="" type="checkbox"/> | Not Deferrable |

Referenced Constraint:

Schema: Table: Constraint: On Delete:

Associations:

| Local Column | Referenced Column |
|--------------|-------------------|
| BOOK_ID | BOOK_ID |

Pomoc OK Anuluj

Kończymy tworzenie tabeli klikając OK.

Możemy teraz podglądnąć w panelu *Connections*, że został utworzony nowy wyzwalacz (w menu *Triggers*) oraz nowa sekwencja (w menu *Sequences*).

Dla porównania poniżej znajduje się polecenie SQL tworzące tabelę TRANSACTIONS (ale nie tworzące dla niej wyzwalacza oraz sekwencji!).

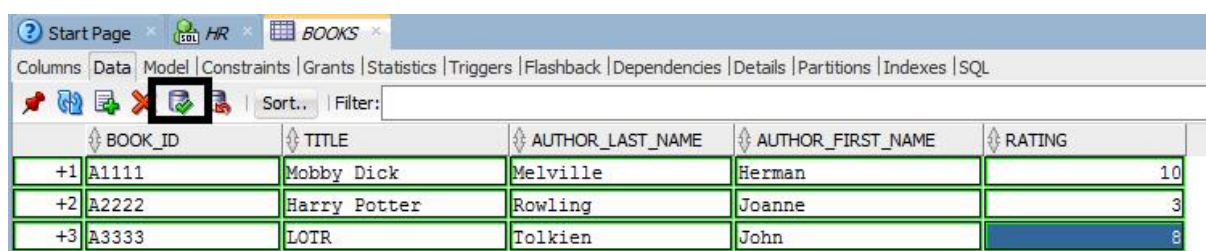
```
CREATE TABLE transactions
```

```
(  
  transaction_id NUMBER,  
  patron_id CONSTRAINT for_key_patron_id REFERENCES patrons(patron_id),  
  book_id CONSTRAINT for_key_book_id REFERENCES books(book_id),  
  transaction_date DATE CONSTRAINT tran_date_not_null NOT NULL,  
  transaction_type NUMBER CONSTRAINT tran_type_not_null NOT NULL,  
  CONSTRAINT transactions_pk PRIMARY KEY (transaction_id)  
);
```

Wprowadzanie danych do tabel

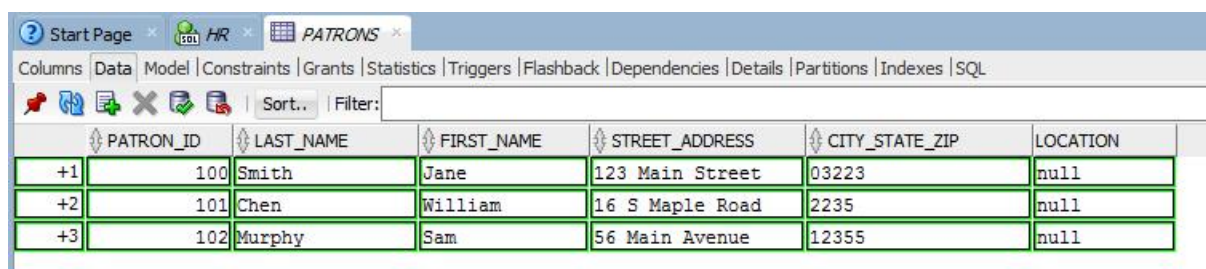
Dane zostaną wprowadzone za pomocą komponentów graficznych środowiska (a nie przy wykorzystaniu poleceń SQL).

Klikamy na tabelę **BOOKS** w menu *Connections*. Następnie wybieramy zakładkę *Data* i za pomocą ikonki *Insert Row* wprowadzamy dane jak na rysunku poniżej. Następnie *zatwierdzamy* wprowadzone dane za pomocą ikony zaznaczonej (czarną ramką) na rysunku.



| | BOOK_ID | TITLE | AUTHOR_LAST_NAME | AUTHOR_FIRST_NAME | RATING |
|----|---------|--------------|------------------|-------------------|--------|
| +1 | A1111 | Moby Dick | Melville | Herman | 10 |
| +2 | A2222 | Harry Potter | Rowling | Joanne | 3 |
| +3 | A3333 | LOTR | Tolkien | John | 8 |

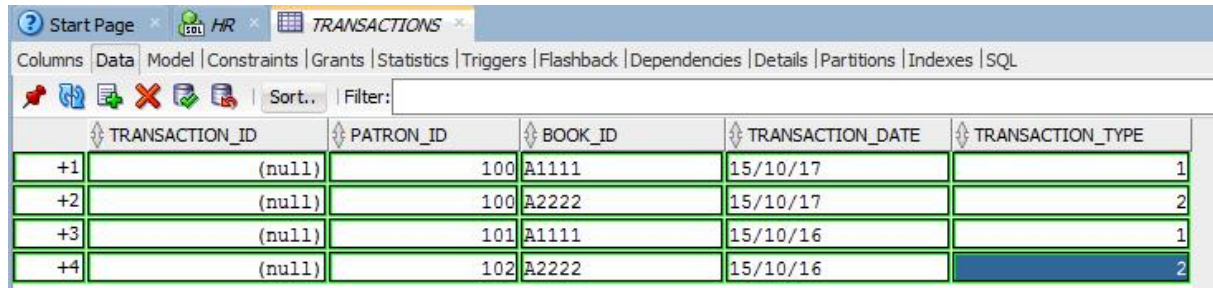
Podobnie wprowadzamy dane dla tabeli **PATRONS**.



| | PATRON_ID | LAST_NAME | FIRST_NAME | STREET_ADDRESS | CITY_STATE_ZIP | LOCATION |
|----|-----------|-----------|------------|-----------------|----------------|----------|
| +1 | 100 | Smith | Jane | 123 Main Street | 03223 | null |
| +2 | 101 | Chen | William | 16 S Maple Road | 2235 | null |
| +3 | 102 | Murphy | Sam | 56 Main Avenue | 12355 | null |

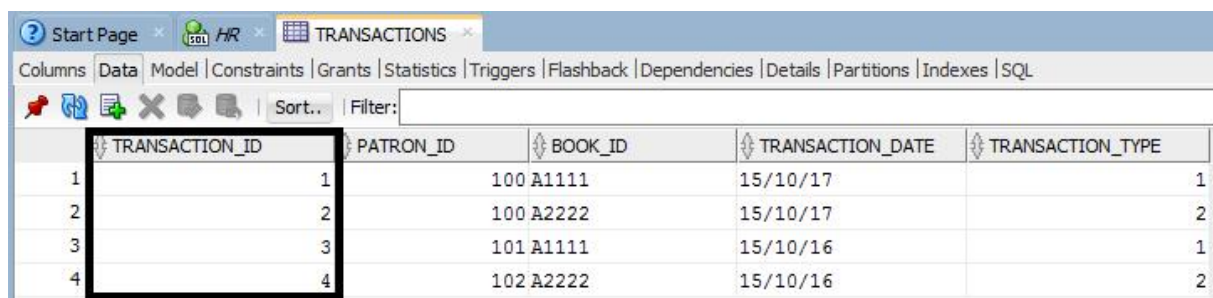
Oraz dla tabeli **TRANSACTIONS**. Przy czym nie ma potrzeby podawania wartości TRANSACTION_ID bo zostanie ona wygenerowana automatycznie. Zauważ, że wartości kolumn PATRON_ID oraz BOOK_ID muszą istnieć już w tabelach BOOKS i PATRONS, inaczej nie będzie możliwe wprowadzenie danych (ze względu na naruszenie ograniczenia klucza obcego).

Bezpośrednio przed zatwierdzeniem danych w tabeli **TRANSACTIONS**:



| | TRANSACTION_ID | PATRON_ID | BOOK_ID | TRANSACTION_DATE | TRANSACTION_TYPE |
|----|----------------|-----------|---------|------------------|------------------|
| +1 | (null) | 100 | A1111 | 15/10/17 | 1 |
| +2 | (null) | 100 | A2222 | 15/10/17 | 2 |
| +3 | (null) | 101 | A1111 | 15/10/16 | 1 |
| +4 | (null) | 102 | A2222 | 15/10/16 | 2 |

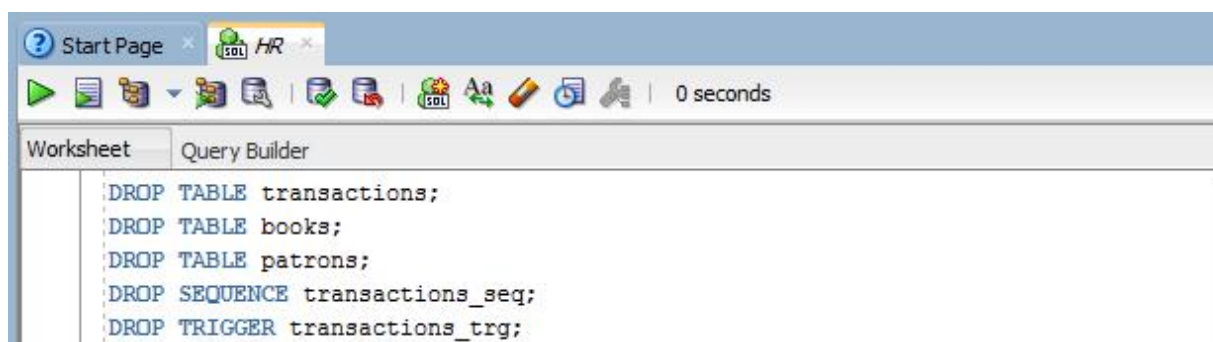
Bezpośrednio po zatwierdzeniu danych (ikonką *Commit Changes*).



| | TRANSACTION_ID | PATRON_ID | BOOK_ID | TRANSACTION_DATE | TRANSACTION_TYPE |
|---|----------------|-----------|---------|------------------|------------------|
| 1 | 1 | 100 | A1111 | 15/10/17 | 1 |
| 2 | 2 | 100 | A2222 | 15/10/17 | 2 |
| 3 | 3 | 101 | A1111 | 15/10/16 | 1 |
| 4 | 4 | 102 | A2222 | 15/10/16 | 2 |

Usunięcie utworzonych tabel i wprowadzonych danych

Na zakończenie ćwiczenia w arkuszu roboczym (*Worksheet*) należy wydać następujące polecenia



```

DROP TABLE transactions;
DROP TABLE books;
DROP TABLE patrons;
DROP SEQUENCE transactions_seq;
DROP TRIGGER transactions_trg;
  
```

