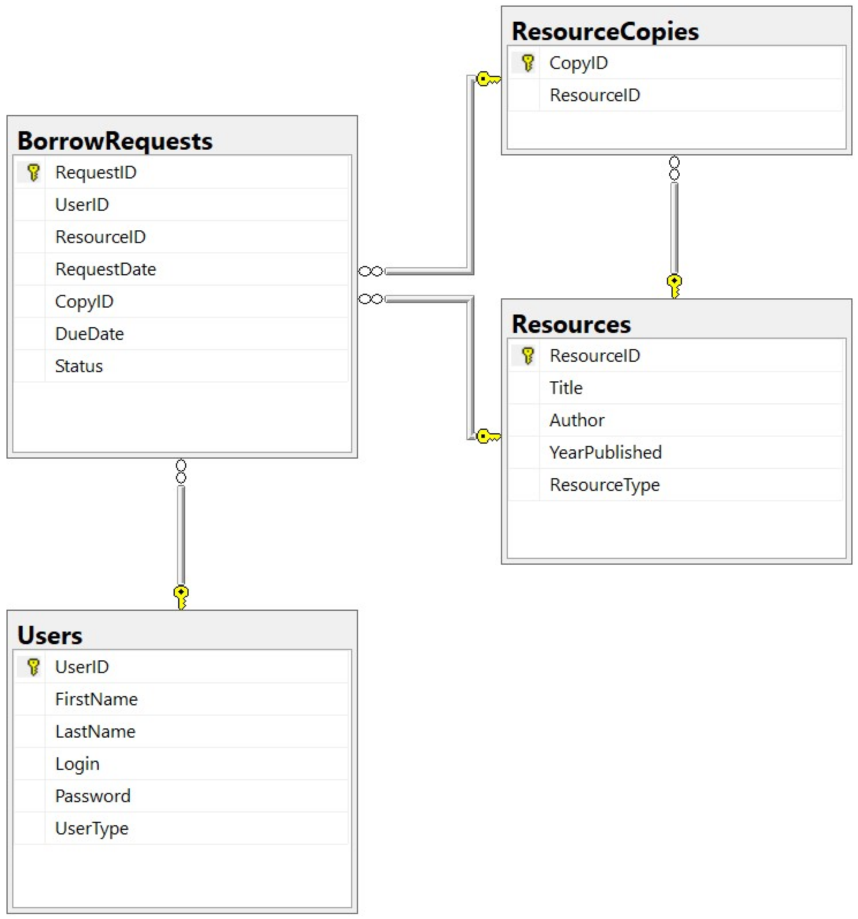
1. Sformułowanie wymagań dotyczących dostępu do bazy i jej zawartości

TABELA\UŻYTKOWNIK	Administrator	Pracownik	Klient
Users	Wszystko	SELECT, UPDATE	SELECT
Resources		SELECT, INSERT, UPDATE, DELETE	SELECT
ResourceCopies		SELECT, INSERT, UPDATE, DELETE	SELECT
BorrowRequests		SELECT, INSERT, UPDATE	SELECT, INSERT, UPDATE

2. Diagram ERD



3. Model fizyczny bazy danych
    a. Tabele

```
--TABLES
CREATE TABLE Users (
    UserID INT PRIMARY KEY IDENTITY(1,1),
    FirstName VARCHAR(255) NOT NULL,
    LastName VARCHAR(255) NOT NULL,
    Login VARCHAR(255) UNIQUE NOT NULL,
    Password VARCHAR(255) NOT NULL,
    UserType VARCHAR(50) NOT NULL -- Client / Employee
);

CREATE TABLE Resources (
    ResourceID INT PRIMARY KEY IDENTITY(1,1),
    Title VARCHAR(255) NOT NULL,
    Author VARCHAR(255) NOT NULL,
    YearPublished INT NOT NULL,
    ResourceType VARCHAR(50) NOT NULL, -- Book / Magazine / ....
);

-- represents individual copies of resources
CREATE TABLE ResourceCopies (
    CopyID INT PRIMARY KEY IDENTITY(1,1),
    ResourceID INT NOT NULL,
    FOREIGN KEY (ResourceID) REFERENCES Resources(ResourceID),
);

CREATE TABLE BorrowRequests (
    RequestID INT PRIMARY KEY IDENTITY(1,1),
    UserID INT NOT NULL,
    ResourceID INT NOT NULL,
    RequestDate DATE NOT NULL,
    CopyID INT, -- Assigned by Employee after acceptance
    DueDate DATE, -- Assigned by Employee after acceptance
    Status VARCHAR(50) NOT NULL, -- Pending / Approved / Returned
    FOREIGN KEY (UserID) REFERENCES Users(UserID),
    FOREIGN KEY (CopyID) REFERENCES ResourceCopies(CopyID),
    FOREIGN KEY (ResourceID) REFERENCES Resources(ResourceID),
);

GO
```

b. Widoki

```sql
CREATE VIEW SummarisedResources AS
SELECT
    R.ResourceID,
    R.Title,
    COUNT(RC.CopyID) AS TotalCopies,
    COUNT(CASE WHEN BR.Status = 'Approved' THEN 1 END) AS BorrowedCopies
FROM Resources R
LEFT JOIN ResourceCopies RC ON R.ResourceID = RC.ResourceID
LEFT JOIN BorrowRequests BR ON RC.CopyID = BR.CopyID
GROUP BY R.ResourceID, R.Title;
GO

CREATE VIEW UserDetailsWithBorrowedResources AS
SELECT
    U.UserID,
    U.FirstName,
    U.LastName,
    U.Login,
    U.UserType,
    BR.RequestID,
    BR.RequestDate,
    BR.DueDate,
    R.Title AS BorrowedResource
FROM Users U
JOIN BorrowRequests BR ON U.UserID = BR.UserID
JOIN ResourceCopies RC ON BR.CopyID = RC.CopyID
JOIN Resources R ON BR.ResourceID = R.ResourceID
WHERE BR.Status = 'Approved';
GO

CREATE VIEW DelayedBorrowersView AS
SELECT
    U.UserID,
    U.FirstName,
    U.LastName,
    BR.RequestID,
    R.Title AS BorrowedResource,
    BR.DueDate,
    DATEDIFF(DAY, BR.DueDate, GETDATE()) AS DaysLate
FROM Users U
JOIN BorrowRequests BR ON U.UserID = BR.UserID
JOIN ResourceCopies RC ON BR.CopyID = RC.CopyID
JOIN Resources R ON BR.ResourceID = R.ResourceID
WHERE BR.Status = 'Returned' AND BR.DueDate < GETDATE();
GO
```

```sql
CREATE VIEW ApprovedBorrowRequests AS
SELECT
    BR.RequestID,
    U.FirstName,
    U.LastName,
    R.Title AS BorrowedResource,
    BR.RequestDate,
    BR.DueDate,
    BR.Status
FROM BorrowRequests BR
JOIN Users U ON BR.UserID = U.UserID
JOIN Resources R ON BR.ResourceID = R.ResourceID
WHERE BR.Status = 'Approved';
GO

CREATE VIEW PendingBorrowRequests AS
SELECT
    BR.RequestID,
    U.FirstName,
    U.LastName,
    R.Title AS RequestedResource,
    BR.RequestDate,
    BR.DueDate
FROM BorrowRequests BR
JOIN Users U ON BR.UserID = U.UserID
JOIN Resources R ON BR.ResourceID = R.ResourceID
WHERE BR.Status = 'Pending';
GO
```

c. Przykładowe dane

```sql
-- Users
INSERT INTO Users (FirstName, LastName, Login, Password, UserType) VALUES
('Adam', 'Nowak', 'anowak', 'pass123', 'Client'),
('Ewa', 'Kowalska', 'ekowalska', 'securepass', 'Client'),
('Mateusz', 'Wójcik', 'mwojcik', 'pass456', 'Client'),
('Karolina', 'Lis', 'klis', 'strongpass', 'Client'),
('Marcin', 'Kaczmarek', 'mkaczmarek', 'userpass', 'Client'),
('Alicja', 'Pawlak', 'apawlak', 'pass789', 'Client'),
('Michał', 'Duda', 'mduda', 'testpass', 'Client'),
('Katarzyna', 'Szymańska', 'kszymanska', 'mypassword', 'Client'),
('Piotr', 'Kowalczyk', 'pkowalczyk', 'pass1234', 'Client'),
('Anna', 'Jankowska', 'ajankowska', 'pass5678', 'Client'),
('Tomasz', 'Wiśniewski', 'twisniewski', 'securepass', 'Client'),
('Magdalena', 'Zając', 'mzajac', 'pass987', 'Client'),
('Grzegorz', 'Kowal', 'gkowal', 'testpass', 'Employee'),
('Agnieszka', 'Nowicka', 'anowicka', 'mypass', 'Employee'),
('Łukasz', 'Sikora', 'lsikora', 'pass654', 'Employee');

-- Resources
INSERT INTO Resources (Title, Author, YearPublished, ResourceType) VALUES
('Database Management', 'John Smith', 2020, 'Book'),
('Web Development Basics', 'Alice Johnson', 2019, 'Book'),
('Data Science in Practice', 'Michael Brown', 2022, 'Magazine'),
('Java Programming', 'Emily Davis', 2021, 'Book'),
('Introduction to Python', 'Daniel Wilson', 2018, 'Magazine'),
('Artificial Intelligence Fundamentals', 'Sophie White', 2020, 'Book'),
('Network Security', 'Andrew Miller', 2019, 'Book'),
('Software Engineering Principles', 'Olivia Taylor', 2022, 'Magazine'),
('Machine Learning Applications', 'William Brown', 2021, 'Book'),
('Mobile App Development', 'Emma Turner', 2018, 'Magazine');
```

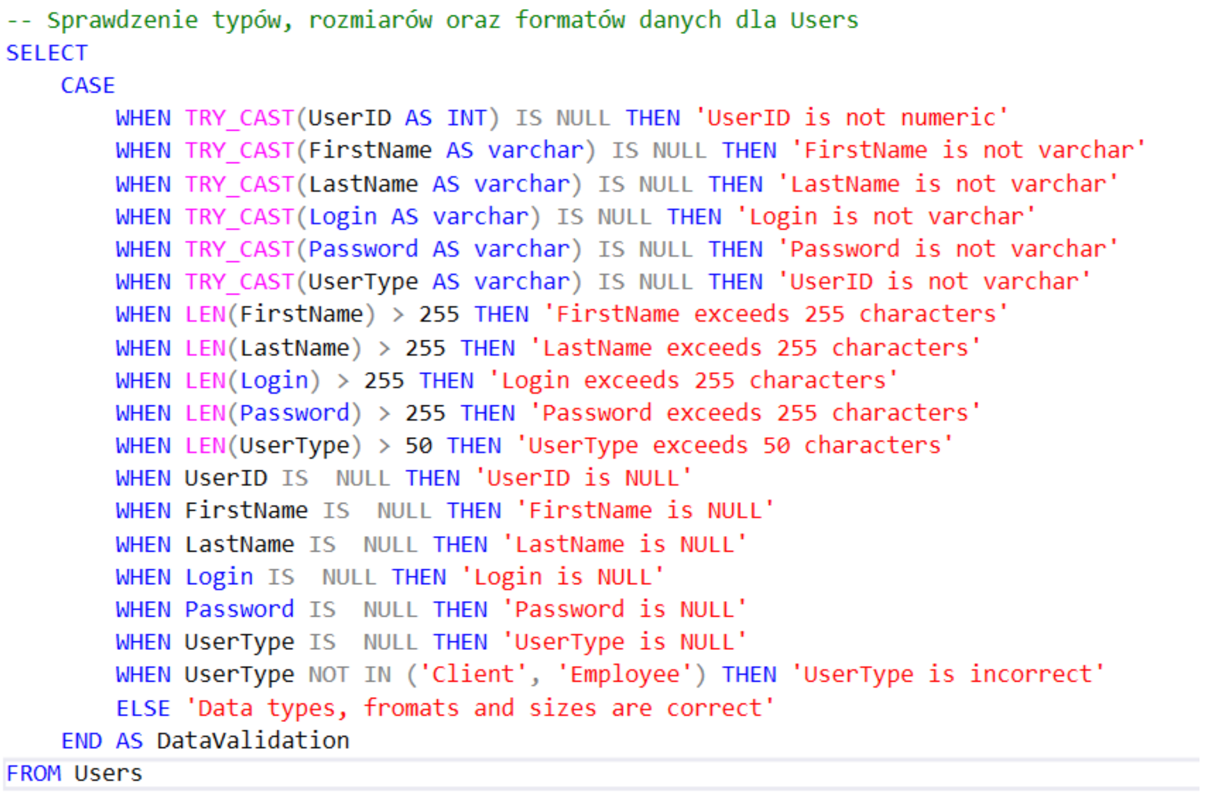
```sql
-- ResourceCopies
INSERT INTO ResourceCopies (ResourceID) VALUES
(1),
(1),
(2),
(3),
(4),
(5),
(6),
(7),
(8),
(9),
(10),
(1),
(2),
(3),
(4),
(5),
(6),
(7),
(8),
(9),
(10),
(1),
(2),
(3),
(4),
(5),
(6),
(7),
(8),
(9),
(10),
(1),
(2),
(3),
(4),
(5),
(6),
(7),
(8),
(9),
(10);
```

```sql
-- BorrowRequests
INSERT INTO BorrowRequests (UserID, ResourceID, RequestDate, CopyID, DueDate,
Status) VALUES
(1, 1, '2023-01-01', 1, '2023-01-15', 'Approved'),
(3, 2, '2023-01-02', 3, '2023-01-16', 'Pending'),
(5, 3, '2023-01-03', NULL, NULL, 'Pending'),
(2, 4, '2023-01-04', NULL, '2023-11-29', 'Approved'),
(4, 5, '2023-01-05', 5, '2023-01-20', 'Returned'),
(6, 6, '2023-01-06', 6, '2023-01-21', 'Approved'),
(8, 7, '2023-01-07', 8, '2023-01-22', 'Pending'),
(10, 8, '2023-01-08', NULL, '2024-01-01', 'Approved'),
(7, 9, '2023-01-09', NULL, NULL, 'Pending'),
(9, 10, '2023-01-10', 10, '2023-01-25', 'Approved'),
(11, 1, '2023-01-11', 11, '2023-01-26', 'Returned'),
(7, 2, '2023-01-12', 12, '2023-01-27', 'Pending'),
(9, 3, '2023-01-13', NULL, NULL, 'Pending'),
(12, 4, '2023-01-14', 13, '2023-01-28', 'Approved'),
(1, 5, '2023-01-15', 14, '2023-01-29', 'Approved'),
(4, 6, '2023-01-16', 15, '2023-01-30', 'Returned'),
(3, 7, '2023-01-17', NULL, '2024-02-01', 'Approved'),
(5, 8, '2023-01-18', 16, '2023-01-31', 'Pending'),
(4, 9, '2023-01-19', 17, '2023-02-01', 'Returned'),
(1, 10, '2023-01-20', 18, '2023-02-02', 'Pending');
```

4. Wdrożenie i przetestowanie bazy

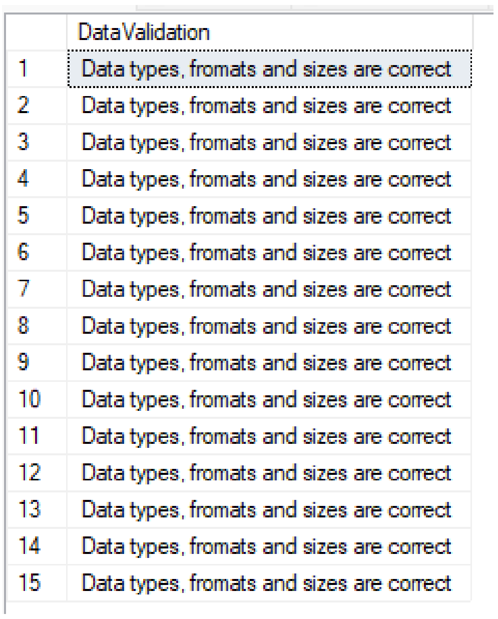
Tabela Users:

Sprawdzenie integralności semantycznej:
Zapytania:



```
-- Sprawdzenie typów, rozmiarów oraz formatów danych dla Users
SELECT
    CASE
        WHEN TRY_CAST(UserID AS INT) IS NULL THEN 'UserID is not numeric'
        WHEN TRY_CAST(FirstName AS varchar) IS NULL THEN 'FirstName is not varchar'
        WHEN TRY_CAST(LastName AS varchar) IS NULL THEN 'LastName is not varchar'
        WHEN TRY_CAST(Login AS varchar) IS NULL THEN 'Login is not varchar'
        WHEN TRY_CAST(Password AS varchar) IS NULL THEN 'Password is not varchar'
        WHEN TRY_CAST(UserType AS varchar) IS NULL THEN 'UserID is not varchar'
        WHEN LEN(FirstName) > 255 THEN 'FirstName exceeds 255 characters'
        WHEN LEN(LastName) > 255 THEN 'LastName exceeds 255 characters'
        WHEN LEN(Login) > 255 THEN 'Login exceeds 255 characters'
        WHEN LEN(Password) > 255 THEN 'Password exceeds 255 characters'
        WHEN LEN(UserType) > 50 THEN 'UserType exceeds 50 characters'
        WHEN UserID IS  NULL THEN 'UserID is NULL'
        WHEN FirstName IS  NULL THEN 'FirstName is NULL'
        WHEN LastName IS  NULL THEN 'LastName is NULL'
        WHEN Login IS  NULL THEN 'Login is NULL'
        WHEN Password IS  NULL THEN 'Password is NULL'
        WHEN UserType IS  NULL THEN 'UserType is NULL'
        WHEN UserType NOT IN ('Client', 'Employee') THEN 'UserType is incorrect'
        ELSE 'Data types, fromats and sizes are correct'
    END AS DataValidation
FROM Users
```

Wyniki:



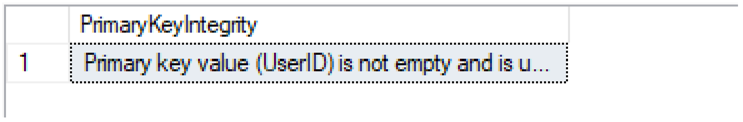
	DataValidation
1	Data types, fromats and sizes are correct
2	Data types, fromats and sizes are correct
3	Data types, fromats and sizes are correct
4	Data types, fromats and sizes are correct
5	Data types, fromats and sizes are correct
6	Data types, fromats and sizes are correct
7	Data types, fromats and sizes are correct
8	Data types, fromats and sizes are correct
9	Data types, fromats and sizes are correct
10	Data types, fromats and sizes are correct
11	Data types, fromats and sizes are correct
12	Data types, fromats and sizes are correct
13	Data types, fromats and sizes are correct
14	Data types, fromats and sizes are correct
15	Data types, fromats and sizes are correct

Sprawdzenie integralności encji:

Zapytania:

```
-- Sprawdzenie niepustości i unikalności wartości klucza głównego (UserID)
SELECT
    CASE
        WHEN COUNT(*) = COUNT(DISTINCT UserID) AND COUNT(UserID) = COUNT(NULLIF(UserID, NULL))
        THEN 'Primary key value (UserID) is not empty, and is unique'
        ELSE 'Primary key value (UserID) is empty, or is not unique'
    END AS PrimaryKeyIntegrity
FROM Users;
```

Wyniki:



	PrimaryKeyIntegrity
1	Primary key value (UserID) is not empty and is u...

Sprawdzenie integralności referencji:

BRAK

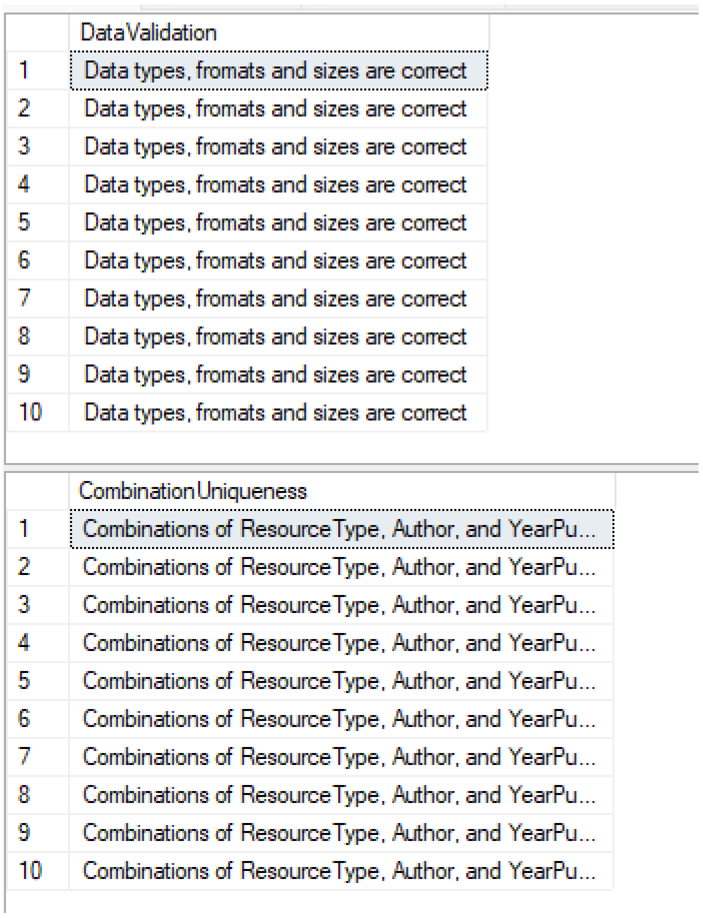
Tabela Resources:

Sprawdzenie integralności semantycznej:
Zapytania:

```
-- Sprawdzenie typów, rozmiarów oraz formatów danych dla Resources
SELECT
    CASE
        WHEN TRY_CAST(ResourceID AS INT) IS NULL THEN 'ResourceID is not numeric'
        WHEN TRY_CAST(Title AS VARCHAR) IS NULL THEN 'Title is not varchar'
        WHEN TRY_CAST(Author AS VARCHAR) IS NULL THEN 'Author is not varchar'
        WHEN TRY_CAST(YearPublished AS INT) IS NULL THEN 'YearPublished is not numeric'
        WHEN TRY_CAST(ResourceType AS VARCHAR) IS NULL THEN 'ResourceType is not varchar'
        WHEN LEN(Title) > 255 THEN 'Title exceeds 255 characters'
        WHEN LEN(Author) > 255 THEN 'Author exceeds 255 characters'
        WHEN LEN(ResourceType) > 50 THEN 'ResourceType exceeds 50 characters'
        WHEN ResourceID IS  NULL THEN 'ResourceID is NULL'
        WHEN Title IS  NULL THEN 'Title is NULL'
        WHEN YearPublished IS  NULL THEN 'YearPublished is NULL'
        WHEN ResourceType IS  NULL THEN 'ResourceType is NULL'
        WHEN ResourceType NOT IN ('Book', 'Magazine','Article','Letter') THEN 'ResourceType is incorrect'
        ELSE 'Data types, fromats and sizes are correct'
    END AS DataValidation
FROM Resources

-- Sprawdzenie, czy dla danego tytułu istnieją różne kombinacje rodzaju zasobu, autora i roku publikacji
SELECT
    CASE
        WHEN COUNT(*) <> (
            SELECT COUNT(DISTINCT ResourceType + '|' + Author + '|' + CAST(YearPublished AS VARCHAR(10)))
            FROM Resources R2
            WHERE R2.Title = R1.Title
        )
        THEN 'Duplicate combinations of ResourceType, Author, and YearPublished found for the same title'
        ELSE 'Combinations of ResourceType, Author, and YearPublished are unique for each title'
    END AS CombinationUniqueness
FROM Resources R1
GROUP BY Title;
```

Wyniki:



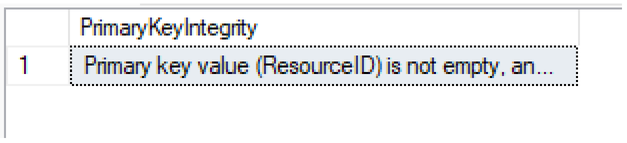
	DataValidation
1	Data types, fromats and sizes are correct
2	Data types, fromats and sizes are correct
3	Data types, fromats and sizes are correct
4	Data types, fromats and sizes are correct
5	Data types, fromats and sizes are correct
6	Data types, fromats and sizes are correct
7	Data types, fromats and sizes are correct
8	Data types, fromats and sizes are correct
9	Data types, fromats and sizes are correct
10	Data types, fromats and sizes are correct

	CombinationUniqueness
1	Combinations of ResourceType, Author, and YearPu...
2	Combinations of ResourceType, Author, and YearPu...
3	Combinations of ResourceType, Author, and YearPu...
4	Combinations of ResourceType, Author, and YearPu...
5	Combinations of ResourceType, Author, and YearPu...
6	Combinations of ResourceType, Author, and YearPu...
7	Combinations of ResourceType, Author, and YearPu...
8	Combinations of ResourceType, Author, and YearPu...
9	Combinations of ResourceType, Author, and YearPu...
10	Combinations of ResourceType, Author, and YearPu...

Sprawdzenie integralności encji:
Zapytania:

```sql
-- Sprawdzenie niepustości i unikalności wartości klucza głównego (ResourceID)
SELECT
    CASE
        WHEN COUNT(*) = COUNT(DISTINCT ResourceID) AND COUNT(ResourceID) = COUNT(NULLIF(ResourceID, NULL))
        THEN 'Primary key value (ResourceID) is not empty, and is unique'
        ELSE 'Primary key value (ResourceID) is empty, or is not unique'
    END AS PrimaryKeyIntegrity
FROM Resources;
```

Wyniki:



	PrimaryKeyIntegrity
1	Primary key value (ResourceID) is not empty, an...

Sprawdzenie integralności referencji:

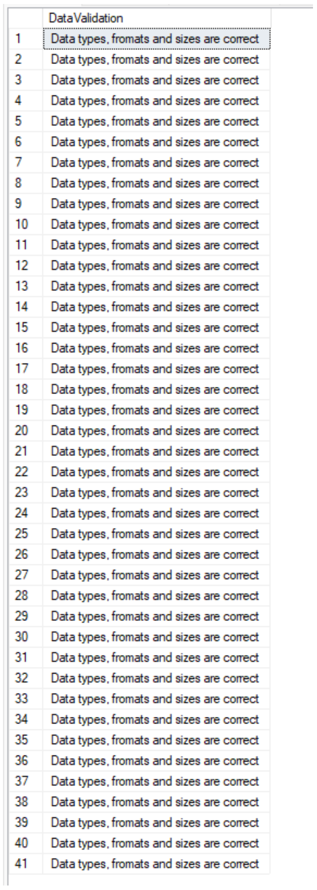
BRAK

Tabela ResourcesCopies

Sprawdzenie integralności semantycznej:
Zapytania:

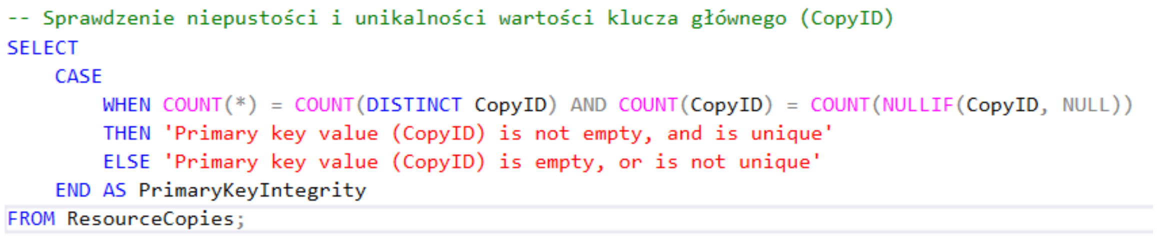
```
-- Sprawdzenie typów, rozmiarów oraz formatów danych dla ResourcesCopies
SELECT
    CASE
        WHEN TRY_CAST(ResourceID AS INT) IS NULL THEN 'ResourceID is not numeric'
        WHEN TRY_CAST(CopyID AS INT) IS NULL THEN 'CopyID is not numeric'
        WHEN ResourceID IS  NULL THEN 'ResourceID is NULL'
        WHEN CopyID IS  NULL THEN 'CopyID is NULL'
        ELSE 'Data types, fromats and sizes are correct'
    END AS DataValidation
FROM ResourceCopies
```

Wyniki:



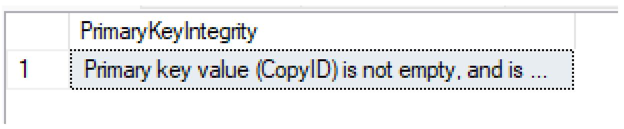
	DataValidation
1	Data types, fromats and sizes are correct
2	Data types, fromats and sizes are correct
3	Data types, fromats and sizes are correct
4	Data types, fromats and sizes are correct
5	Data types, fromats and sizes are correct
6	Data types, fromats and sizes are correct
7	Data types, fromats and sizes are correct
8	Data types, fromats and sizes are correct
9	Data types, fromats and sizes are correct
10	Data types, fromats and sizes are correct
11	Data types, fromats and sizes are correct
12	Data types, fromats and sizes are correct
13	Data types, fromats and sizes are correct
14	Data types, fromats and sizes are correct
15	Data types, fromats and sizes are correct
16	Data types, fromats and sizes are correct
17	Data types, fromats and sizes are correct
18	Data types, fromats and sizes are correct
19	Data types, fromats and sizes are correct
20	Data types, fromats and sizes are correct
21	Data types, fromats and sizes are correct
22	Data types, fromats and sizes are correct
23	Data types, fromats and sizes are correct
24	Data types, fromats and sizes are correct
25	Data types, fromats and sizes are correct
26	Data types, fromats and sizes are correct
27	Data types, fromats and sizes are correct
28	Data types, fromats and sizes are correct
29	Data types, fromats and sizes are correct
30	Data types, fromats and sizes are correct
31	Data types, fromats and sizes are correct
32	Data types, fromats and sizes are correct
33	Data types, fromats and sizes are correct
34	Data types, fromats and sizes are correct
35	Data types, fromats and sizes are correct
36	Data types, fromats and sizes are correct
37	Data types, fromats and sizes are correct
38	Data types, fromats and sizes are correct
39	Data types, fromats and sizes are correct
40	Data types, fromats and sizes are correct
41	Data types, fromats and sizes are correct

Sprawdzenie integralności encji:
Zapytania:



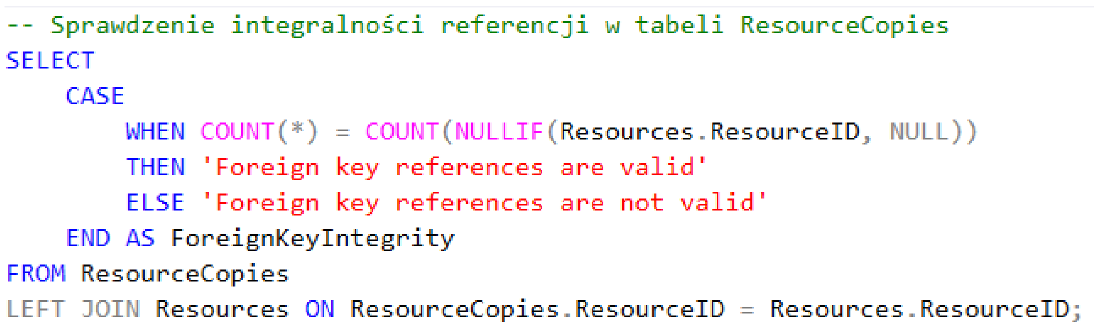
```
-- Sprawdzenie niepustości i unikalności wartości klucza głównego (CopyID)
SELECT
    CASE
        WHEN COUNT(*) = COUNT(DISTINCT CopyID) AND COUNT(CopyID) = COUNT(NULLIF(CopyID, NULL))
        THEN 'Primary key value (CopyID) is not empty, and is unique'
        ELSE 'Primary key value (CopyID) is empty, or is not unique'
    END AS PrimaryKeyIntegrity
FROM ResourceCopies;
```

Wyniki:



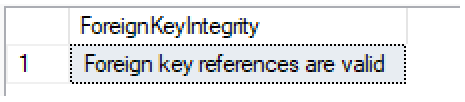
	PrimaryKeyIntegrity
1	Primary key value (CopyID) is not empty, and is ...

Sprawdzenie integralności referencji:
Zapytania:



```
-- Sprawdzenie integralności referencji w tabeli ResourceCopies
SELECT
    CASE
        WHEN COUNT(*) = COUNT(NULLIF(Resources.ResourceID, NULL))
        THEN 'Foreign key references are valid'
        ELSE 'Foreign key references are not valid'
    END AS ForeignKeyIntegrity
FROM ResourceCopies
LEFT JOIN Resources ON ResourceCopies.ResourceID = Resources.ResourceID;
```

Wyniki:



	ForeignKeyIntegrity
1	Foreign key references are valid

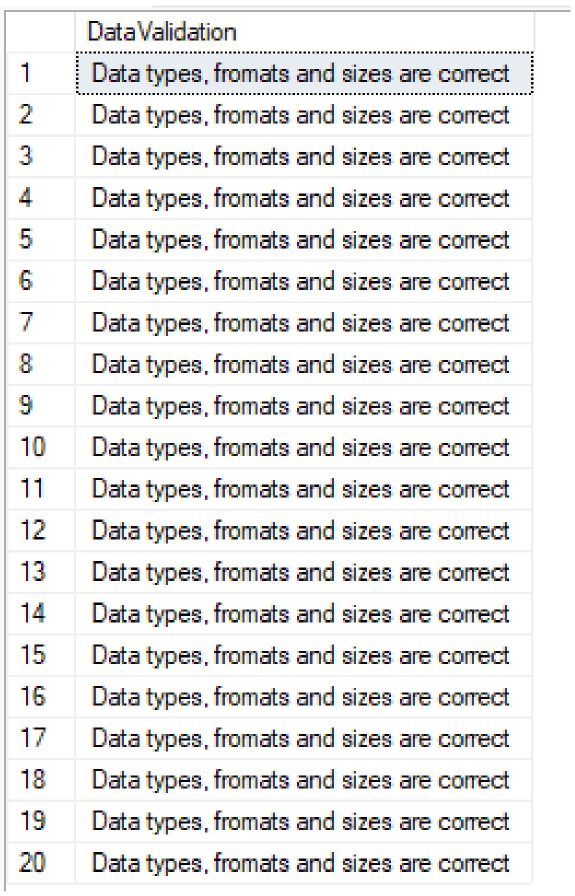
Tabela BorrowRequests:

Sprawdzenie integralności semantycznej:

Zapytania:

```sql
-- Sprawdzenie typów, rozmiarów oraz formatów danych dla BorrowRequests
SELECT
    CASE
        WHEN TRY_CAST(RequestID AS INT) IS NULL THEN 'RequestID is not numeric'
        WHEN TRY_CAST(UserID AS INT) IS NULL THEN 'UserID is not numeric'
        WHEN TRY_CAST(ResourceID AS INT) IS NULL THEN 'ResourceID is not numeric'
        WHEN TRY_CAST(RequestDate AS DATE) IS NULL THEN 'RequestDate is not a valid date'
        WHEN TRY_CAST(CopyID AS INT) IS NULL AND CopyID IS NOT NULL THEN 'CopyID is not numeric'
        WHEN TRY_CAST(DueDate AS DATE) IS NULL AND DueDate IS NOT NULL THEN 'DueDate is not a valid date'
        WHEN TRY_CAST(Status AS VARCHAR) IS NULL THEN 'Status is not varchar'
        WHEN LEN(Status) > 50 THEN 'Status exceeds 50 characters'
        WHEN RequestID IS NULL THEN 'RequestID is NULL'
        WHEN UserID IS NULL THEN 'UserID is NULL'
        WHEN ResourceID IS NULL THEN 'ResourceID is NULL'
        WHEN RequestDate IS NULL THEN 'RequestDate is NULL'
        WHEN Status NOT IN ('Pending', 'Approved', 'Returned') THEN 'Status is incorrect'
        ELSE 'Data types, fromats and sizes are correct'
    END AS DataValidation
FROM BorrowRequests;
```

Wyniki:

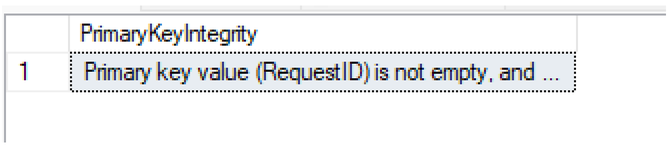


	DataValidation
1	Data types, fromats and sizes are correct
2	Data types, fromats and sizes are correct
3	Data types, fromats and sizes are correct
4	Data types, fromats and sizes are correct
5	Data types, fromats and sizes are correct
6	Data types, fromats and sizes are correct
7	Data types, fromats and sizes are correct
8	Data types, fromats and sizes are correct
9	Data types, fromats and sizes are correct
10	Data types, fromats and sizes are correct
11	Data types, fromats and sizes are correct
12	Data types, fromats and sizes are correct
13	Data types, fromats and sizes are correct
14	Data types, fromats and sizes are correct
15	Data types, fromats and sizes are correct
16	Data types, fromats and sizes are correct
17	Data types, fromats and sizes are correct
18	Data types, fromats and sizes are correct
19	Data types, fromats and sizes are correct
20	Data types, fromats and sizes are correct

Sprawdzenie integralności encji:
Zapytania:

```sql
-- Sprawdzenie niepustości i unikalności wartości klucza głównego (RequestID) w BorrowRequests
SELECT
    CASE
        WHEN COUNT(*) = COUNT(DISTINCT RequestID) AND COUNT(RequestID) = COUNT(NULLIF(RequestID, NULL))
        THEN 'Primary key value (RequestID) is not empty, and is unique'
        ELSE 'Primary key value (RequestID) is empty, or is not unique'
    END AS PrimaryKeyIntegrity
FROM BorrowRequests;
```

Wyniki:



	PrimaryKeyIntegrity
1	Primary key value (RequestID) is not empty, and ...

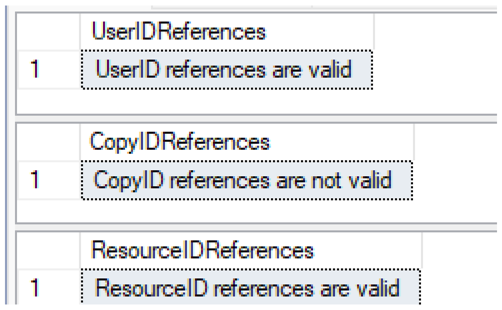
Sprawdzenie integralności referencji:
Zapytania:

```sql
-- Sprawdzenie, czy UserID istnieje w tabeli Users
SELECT
    CASE
        WHEN COUNT(*) <> COUNT(NULLIF(Users.UserID, NULL))
        THEN 'UserID references are not valid'
        ELSE 'UserID references are valid'
    END AS UserIDReferences
FROM BorrowRequests
LEFT JOIN Users ON BorrowRequests.UserID = Users.UserID;

-- Sprawdzenie, czy CopyID istnieje w tabeli ResourceCopies
SELECT
    CASE
        WHEN COUNT(*) <> COUNT(NULLIF(ResourceCopies.CopyID, NULL))
        THEN 'CopyID references are not valid'
        ELSE 'CopyID references are valid'
    END AS CopyIDReferences
FROM BorrowRequests
LEFT JOIN ResourceCopies ON BorrowRequests.CopyID = ResourceCopies.CopyID;

-- Sprawdzenie, czy ResourceID istnieje w tabeli Resources
SELECT
    CASE
        WHEN COUNT(*) <> COUNT(NULLIF(Resources.ResourceID, NULL))
        THEN 'ResourceID references are not valid'
        ELSE 'ResourceID references are valid'
    END AS ResourceIDReferences
FROM BorrowRequests
LEFT JOIN Resources ON BorrowRequests.ResourceID = Resources.ResourceID;
```

Wyniki:



```
   UserIDReferences
1  UserID references are valid

   CopyIDReferences
1  CopyID references are not valid

   ResourceIDReferences
1  ResourceID references are valid
```