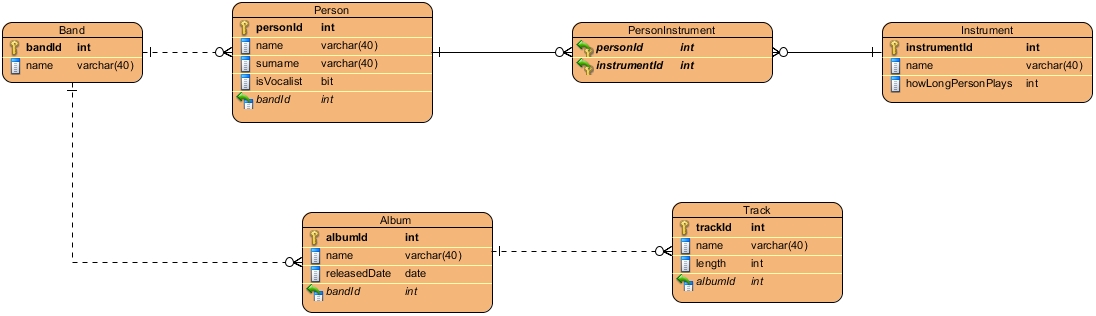
1. Model bazy danych.



1. Zdefiniowanie aspektu związanego z rozproszeniem i przetwarzaniem zasobów

Strategia FAR zostanie zrealizowana w następujący sposób:

1. Fragmentacja

Zostanie zastosowana **fragmentacja** **pozioma**. Zakres tematyczny bazy pozwala na rozdzielenie danych na dwie części. Pierwsza część będzie zawierała zespoły o id mniejszym lub równym 100 oraz powiązane z nią inne tabele. Druga część będzie zawierała dane dotyczące zespołów o id większym niż 100 oraz odpowiadające im tabele. Na podstawie przyjętego założenia można zauważyć, że schemat bazy danych w pierwszym węźle będzie identyczny jak w drugim. W przypadku rozszerzania bazy o kolejne węzły schemat ten będzie również taki sam.

1. Alokacja

W projekcie rozproszenie będzie polegać na podziale danych (według opisanego w poprzednim punkcie sposobu) w dwóch węzłach. Wykorzystywaną bazą danych będzie Microsoft SQL Server. Dla środowiska pierwotnego będzie to Microsoft SQL Server 2019, zaś dla drugiego węzła, który będzie działał w obrębie maszyny wirtualnej będzie to Microsoft SQL Server 2012 Środowisko pierwotne oprócz funkcji zarządzania rozproszeniem będzie również scalało dane pochodzące z drugiego węzła. Druga z instancji (znajdująca się na maszynie wirtualnej) będzie widoczna z instancji znajdującej się w środowisku pierwotnym przy zastosowaniu obiektu serwera (*Server Object*) o nazwie Linked Server (DESKTOP-P8PMK66\MSSQLSERVER2). W celu ułatwienia operowania na danych z drugiej instancji utworzono odpowiednie synonimy, które pozwalają na skrócenie zapisu odwołującego się do Linked Server’a. Kod tworzący synonimy przedstawiono poniżej:

CREATE SYNONYM [dbo].[Album2] FOR [DESKTOP-P8PMK66\MSSQLSERVER2].[MusicShop].[dbo].[Album];

CREATE SYNONYM [dbo].[Band2] FOR [DESKTOP-P8PMK66\MSSQLSERVER2].[MusicShop].[dbo].[Band];

CREATE SYNONYM [dbo].[Instrument2] FOR [DESKTOP-P8PMK66\MSSQLSERVER2].[MusicShop].[dbo].[Instrument];

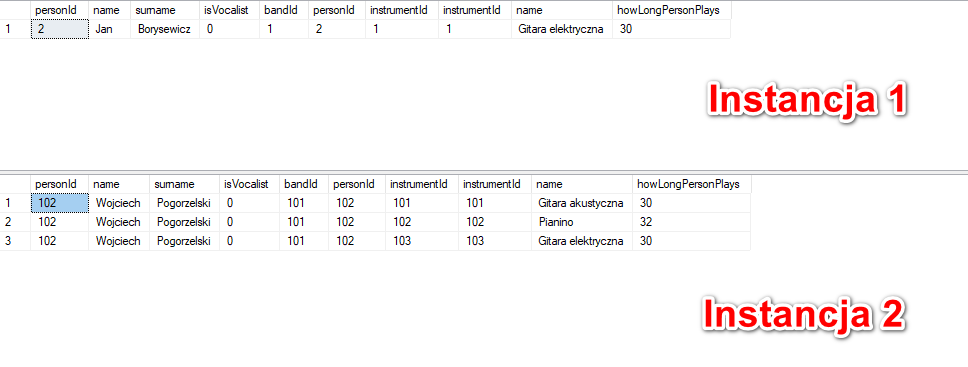
CREATE SYNONYM [dbo].[Person2] FOR [DESKTOP-P8PMK66\MSSQLSERVER2].[MusicShop].[dbo].[Person];

CREATE SYNONYM [dbo].[PersonInstrument2] FOR [DESKTOP-P8PMK66\MSSQLSERVER2].[MusicShop].[dbo].[PersonInstrument];

CREATE SYNONYM [dbo].[Track2] FOR [DESKTOP-P8PMK66\MSSQLSERVER2].[MusicShop].[dbo].[Track];

1. Replikacja

W bazie danych będzie występowała replikacja z redundancją. Redundancja będzie dotyczyła jednej tabeli – Instrument (w pozostałych tabelach będzie występowała replikacja bez redundancji). Dotyczy to sytuacji, gdy osoba wchodząca w skład zespołu, którego dane znajdują się w pierwszej instancji będzie grała na instrumentach, których dane są umieszczone w instancji pierwszej. Przykład został pokazany na poniższym rysunku – instrument o nazwie Gitara elektryczna występuje w dwóch instacjach.



1. Utworzone zapytania:
2. Procedura wyszukująca zespoły, które nie wydały jeszcze albumu:

CREATE PROCEDURE getAllBandsWithoutAlbum AS

SELECT b1.[bandId], b1.[name] FROM [dbo].[Band] b1

LEFT JOIN [dbo].[Album] a1 ON b1.bandId = a1.bandId

WHERE a1.bandId IS NULL

UNION

SELECT b2.[bandId], b2.[name] FROM [dbo].[Band2] b2

LEFT JOIN [dbo].[Album2] a2 ON b2.bandId = a2.bandId

WHERE a2.bandId IS NULL;

1. Procedura wyszukująca osoby, które są multiinstrumentalistami (grają na więcej niż 1 instrumencie)

CREATE PROCEDURE getAllMiltiinstrumentalists AS

SELECT p1.[personId], p1.[name], p1.[surname]

FROM [dbo].[Person] p1

INNER JOIN [dbo].[PersonInstrument] pin1 ON p1.personId=pin1.personId

INNER JOIN [dbo].[Instrument] i1 ON pin1.instrumentId = i1.instrumentId

GROUP BY p1.[personId], p1.[name], p1.[surname]

HAVING COUNT(\*) > 1

UNION

SELECT p2.[personId], p2.[name], p2.[surname]

FROM [dbo].[Person2] p2

INNER JOIN [dbo].[PersonInstrument2] pin2 ON p2.personId=pin2.personId

INNER JOIN [dbo].[Instrument2] i2 ON pin2.instrumentId = i2.instrumentId

GROUP BY p2.[personId], p2.[name], p2.[surname]

HAVING COUNT(\*) > 1;

1. Procedura wyszukująca wszystkich wokalistów

CREATE PROCEDURE getAllVocalists AS

SELECT p1.[personId], p1.[name], p1.[surname]

FROM [dbo].[Person] p1

WHERE p1.isVocalist = 1

UNION

SELECT p2.[personId], p2.[name], p2.[surname]

FROM [dbo].[Person2] p2

WHERE p2.isVocalist = 1

1. Podsumowanie wykonanej pracy

* Zalety oraz wady

Zaletą rozwiązania jest przezroczystość – dane przechowywane są w dwóch instancjach, a pomimo to użytkownik wykonujący procedurę ma wrażenie pracy z jedną instancją bazy danych. Co więcej, zaprojektowane rozwiązanie wpływa na bezpieczeństwo – w przypadku awarii jednego z węzłów tracimy część danych, a nie całość (tj. w przypadku scentralizowanej bazy danych).

Wadą rozwiązania jest natomiast zdefiniowana replikacja z redundancją, która może w przyszłości generować problemy np. z obliczaniem statystyk, które bez uwzględnienia wspominanej redundancji mogą powodować przekłamania.

* Napotkane problemy i rozwiązania

Problemem było połączenie ze sobą instancji bazy danych w środowisku pierwotnym oraz w środowisku wirtualnym (instancja ze środowiska pierwotnego „nie widziała” instancji ze środowiska wirtualnego). Było to spowodowane dwoma czynnikami - włączonym firewall’em po obydwu stronach oraz wyłączonej możliwości komunikacji przez protokół TCP/IP w instancji na maszynie wirtualnej. Problemy zostały rozwiązane na podstawie analizy rozwiązań podobnych problemów w Internecie.

* Pomysły/idee

Podstawowym pomysłem wynikającym z implementacji jest stworzenie przez twórców MS SQL kreatora(Wizarda), który pomógłby rozproszyć bazę danych i monitorować podłączone instancje.

* Subiektywna ocena wybranej bazy

Baza danych MS SQL jest narzędziem, które umożliwiło w sposób wystarczający wykonanie zadania. Przydatnym dodatkiem byłby specjalny kreator omówiony w poprzednim punkcie. Niewątpliwą zaletą MS SQL jest łatwe nawigowanie po strukturze instancji/baz/tabel oraz intuicyjność narzędzia.

Link do repozytorium GitHub: <https://github.com/JakubKap/Rozproszone-bazy-danych>

Link do filmiku z działania aplikacji: <https://www.youtube.com/watch?v=VfTkhqB9dSw>