System Zarządzania Bazą Danych Prywatnej Szkoły Podstawowej

Rozproszony System Bazy Danych

Autorzy: Piotr Ledwoch 245859, Kacper Woźnica 245963
Data: 07.07.2025
Przedmiot: Rozproszone Bazy Danych

1. Wprowadzenie i cel projektu

1.1 Cel projektu

Głównym celem projektu było stworzenie kompleksowego systemu zarządzania bazą danych prywatnej szkoły podstawowej, który integruje różne technologie bazodanowe w architekturze rozproszonej. System ma na celu efektywne zarządzanie wszystkimi aspektami działalności szkoły - od danych uczniów i nauczycieli, przez system ocen i frekwencji, po aspekty finansowe oraz system uwag i komentarzy.

Projekt stanowi demonstrację zaawansowanych technik pracy z rozproszonymi bazami danych, w tym:

- Integracji heterogenicznych systemów bazodanowych
- Replikacji transakcyjnej danych
- Implementacji linked servers
- Zarządzania transakcjami rozproszonymi
- Eksportu danych do systemów zewnętrznych

1.2 Zakres funkcjonalny

System obsługuje następujące obszary funkcjonalne:

1.2.1 Zarządzanie danymi uczniów i nauczycieli

- **Ewidencja uczniów**: przechowywanie danych osobowych, przypisanie do grup klasowych, informacje o rodzicach
- **Zarządzanie nauczycielami**: dane personalne, specjalizacje, przypisanie do przedmiotów i klas
- **Struktura organizacyjna**: grupy klasowe, sale lekcyjne, rozkłady lekcji

1.2.2 System ocen i frekwencji

- **Wystawianie ocen**: oceny z różnych przedmiotów z możliwością dodawania komentarzy i wag
- **Kontrola frekwencji**: rejestracja obecności na lekcjach, generowanie statystyk frekwencji
- **Raportowanie**: szczegółowe raporty postępów uczniów, statystyki klasowe

1.2.3 System finansowy

- **Kontrakty**: zarządzanie umowami z rodzicami dotyczącymi opłat szkolnych
- **Płatności**: obsługa wpłat, monitorowanie zaległości, generowanie przypomnień
- **Rozliczenia**: automatyczne generowanie zobowiązań miesięcznych

1.2.4 System uwag i komentarzy

- **Uwagi pedagogiczne**: możliwość dodawania uwag przez nauczycieli
- **Komentarze**: pozytywne i negatywne komentarze dotyczące zachowania uczniów
- **Historia**: pełna dokumentacja uwag z datami i autorami

1.2.5 Raportowanie i analiza

- **Raporty akademickie**: średnie ocen, statystyki frekwencji, postępy uczniów
- **Raporty finansowe**: stan należności, płatności, zaległości
- **Eksport danych**: możliwość eksportu do formatów Excel dla dalszej analizy

1.3 Założenia projektowe

1.3.1 Założenia architektoniczne

- **Architektura rozproszona**: system składa się z trzech niezależnych baz danych
- **Heterogeniczność**: wykorzystanie różnych systemów bazodanowych (MS SQL Server, Oracle, PostgreSQL)

- **Modularność**: każdy komponent odpowiada za określony obszar funkcjonalny
- **Skalowalność**: możliwość łatwego rozszerzania systemu o nowe funkcjonalności

1.3.2 Założenia technologiczne

- **MS SQL Server**: główna baza danych z danymi uczniów, nauczycieli, ocen i frekwencji
- **Oracle**: wyspecjalizowana baza finansowa z kontraktami i płatnościami
- **PostgreSQL**: system uwag i komentarzy pedagogicznych
- **Linked servers**: technologia integracji umożliwiająca rozproszone zapytania
- **Replikacja**: automatyczna synchronizacja kluczowych danych między serwerami

1.3.3 Założenia bezpieczeństwa

- **Kontrola dostępu**: różne poziomy uprawnień dla różnych typów użytkowników
- **Zabezpieczenia komunikacji**: szyfrowane połączenia między serwerami
- **Audyting**: logowanie wszystkich operacji modyfikujących dane
- **Backup i recovery**: mechanizmy kopii zapasowych dla wszystkich komponentów

1.3.4 Założenia funkcjonalne

- **Integralność danych**: zapewnienie spójności danych między różnymi systemami
- **Transakcje rozproszone**: operacje obejmujące wiele baz danych wykonywane atomowo
- **Wydajność**: optymalizacja zapytań i indeksów dla często wykonywanych operacji
- **Użyteczność**: intuicyjne interfejsy dla różnych grup użytkowników

1.4 Korzyści z implementacji

Zrealizowany system przynosi następujące korzyści:

1. **Centralizacja danych**: wszystkie informacje o działalności szkoły w jednym systemie
2. **Automatyzacja procesów**: automatyczne generowanie raportów, naliczanie należności
3. **Bezpieczeństwo**: kontrolowany dostęp do danych według ról użytkowników
4. **Elastyczność**: możliwość łatwego dostosowywania do zmieniających się wymagań
5. **Skalowalność**: architektura umożliwiająca rozszerzanie systemu
6. **Integracja**: płynne współdziałanie różnych komponentów systemu

System stanowi podstawę dla zarządzania prywatną szkołą podstawową, zapewniając wszystkie niezbędne funkcjonalności przy zachowaniu wysokiej jakości i bezpieczeństwa danych.

2. Architektura systemu

2.1 Ogólna architektura

System zarządzania bazą danych szkoły podstawowej został zaprojektowany w architekturze rozproszonej, składającej się z czterech głównych komponentów współpracujących ze sobą przez standardowe protokoły komunikacyjne. Architektura została zaprojektowana z myślą o modularności, skalowalności oraz łatwości zarządzania różnymi aspektami działalności szkoły.

2.2 Komponenty systemu

2.2.1 MS SQL Server - Serwer główny (SchoolDB)

Lokalizacja: Serwer główny
Baza danych: SchoolDB
Rola: Centralna baza danych systemu

Odpowiada za:

- Przechowywanie danych uczniów, nauczycieli, rodziców
- Zarządzanie strukturą organizacyjną (grupy, sale, przedmioty)
- System ocen i zarządzanie frekwencją
- Koordynację połączeń z innymi komponentami
- Główne procedury CRUD dla encji podstawowych

2.2.2 Oracle Database - Serwer finansowy (FINANCE_DB)

Lokalizacja: Dedykowany serwer Oracle
Schemat: FINANCE_DB
Rola: Wyspecjalizowany system finansowy

Odpowiada za:

- Zarządzanie kontraktami z rodzicami
- Obsługę płatności i należności
- Generowanie zobowiązań miesięcznych
- Raportowanie finansowe
- Integrację z systemem bankowym (symulowana)

2.2.3 PostgreSQL - Serwer uwag (remarks_main)

Lokalizacja: Dedykowany serwer PostgreSQL
Schemat: remarks_main
Rola: System uwag i komentarzy pedagogicznych

Odpowiada za:

- Przechowywanie uwag nauczycieli
- Komentarze dotyczące zachowania uczniów
- System oceny postępów w zachowaniu
- Historię interwencji pedagogicznych

2.2.4 MS SQL Server - Serwer replik (MSSQL_REPLICA)

Lokalizacja: Oddzielna instancja MS SQL Server
Baza danych: SchoolDB_Replica
Rola: Backup i replikacja danych

Odpowiada za:

- Replikację kluczowych danych uczniów
- Zapewnienie ciągłości działania systemu
- Backup operacyjny w czasie rzeczywistym

2.3 Podział funkcjonalny

2.3.1 Warstwa danych podstawowych (MS SQL Server)

```
Tabele główne:
├── students (uczniowie)
├── teachers (nauczyciele)
├── parents (rodzice)
├── groups (grupy klasowe)
├── subjects (przedmioty)
├── marks (oceny)
├── attendances (frekwencja)
└── lessons (lekcje)
```

2.3.2 Warstwa finansowa (Oracle)

```
Tabele finansowe:
├── contracts (kontrakty)
├── payments (płatności)
├── contracts_remote (kontrakty zdalne)
└── payment_summary (podsumowania płatności)
```

2.3.3 Warstwa uwag (PostgreSQL)

```
Tabele uwag:
└── remark (uwagi pedagogiczne)
```

2.4 Komunikacja między komponentami

2.4.1 Linked Servers

System wykorzystuje mechanizm Linked Servers dla komunikacji między heterogenicznymi bazami danych:

- **ORACLE_FINANCE**: Połączenie MS SQL → Oracle
- **POSTGRES_REMARKS**: Połączenie MS SQL → PostgreSQL
- **MSSQL_REPLICA**: Połączenie MS SQL → MS SQL (replikacja)
- **EXCEL_DATA**: Połączenie do plików Excel dla eksportu

2.4.2 Typy komunikacji

Zapytania rozproszone: Umożliwiają wykonywanie zapytań obejmujących dane z wielu serwerów jednocześnie

Transakcje rozproszone: Gwarantują atomowość operacji wykonywanych na wielu bazach danych

Replikacja transakcyjna: Automatyczna synchronizacja danych między serwerem głównym a repliką

Procedury zdalne: Wywołanie procedur składowanych na zdalnych serwerach

2.5 Przepływ danych

2.5.1 Rejestracja nowego ucznia

1. **MS SQL Server**: Utworzenie rekordu ucznia w tabeli students
2. **MS SQL Server**: Powiązanie z rodzicem w tabeli parents_students
3. **Oracle**: Automatyczne utworzenie kontraktu finansowego
4. **Replikacja**: Synchronizacja danych ucznia do serwera repliki

2.5.2 Dodanie uwagi pedagogicznej

1. **MS SQL Server**: Weryfikacja uprawnień nauczyciela
2. **PostgreSQL**: Zapisanie uwagi w tabeli remark
3. **MS SQL Server**: Aktualizacja logów systemowych

2.5.3 Przetworzenie płatności

1. **Oracle**: Identyfikacja zaległej płatności
2. **Oracle**: Aktualizacja statusu płatności
3. **Oracle**: Synchronizacja z tabelą podsumowań
4. **MS SQL Server**: Generowanie powiadomienia (opcjonalne)

2.6 Zalety architektury rozproszonej

2.6.1 Separacja odpowiedzialności

- Każdy komponent zarządza określonym obszarem funkcjonalnym
- Możliwość niezależnego rozwoju i optymalizacji każdego modułu
- Łatwiejsza diagnostyka i rozwiązywanie problemów

2.6.2 Skalowalność pozioma

- Możliwość dodawania nowych serwerów dla różnych funkcjonalności
- Rozproszenie obciążenia między różne instancje
- Możliwość wykorzystania różnych technologii dla różnych zadań

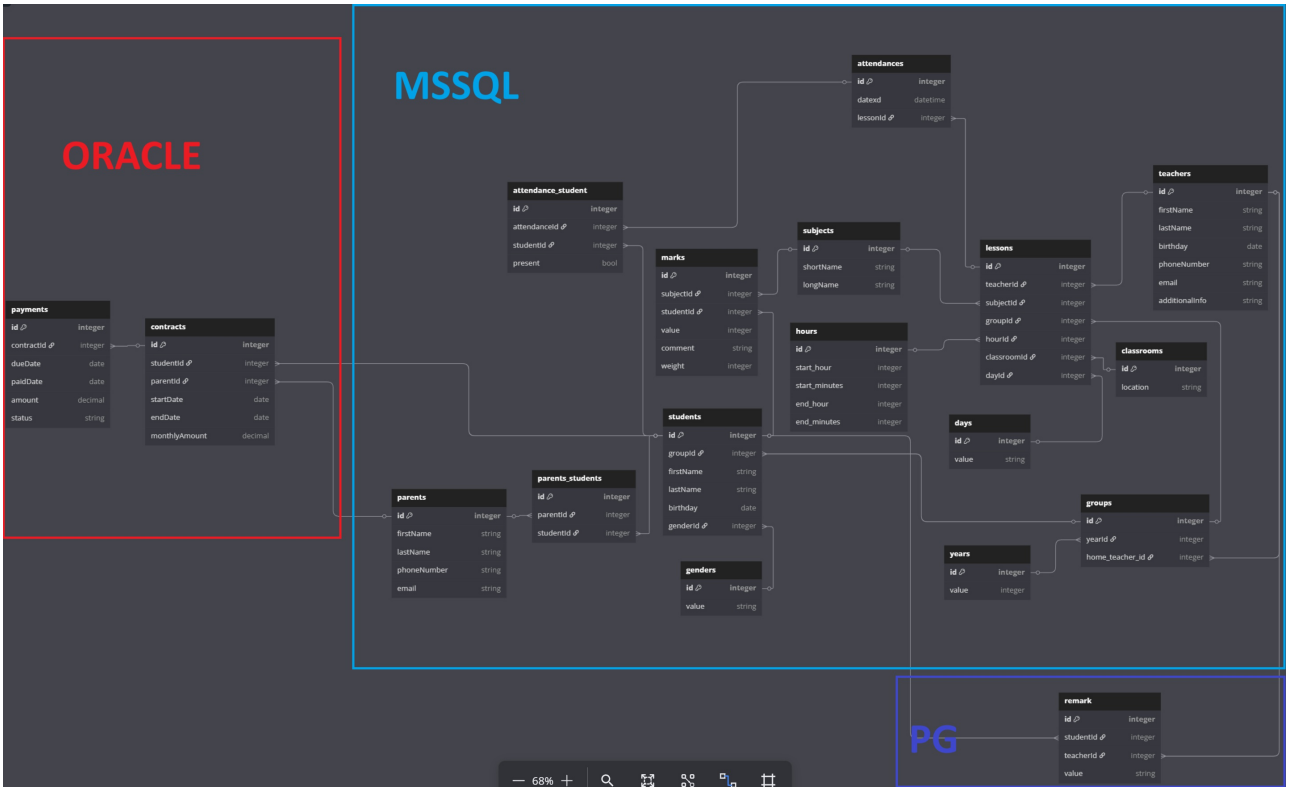
2.6.3 Odporność na awarie

- Awaria jednego komponentu nie powoduje zatrzymania całego systemu
- Replikacja zapewnia ciągłość krytycznych danych
- Możliwość szybkiego przywracania usług

2.6.4 Bezpieczeństwo

- Izolacja danych wrażliwych (np. finansowych) w dedykowanych systemach
- Możliwość implementacji różnych poziomów zabezpieczeń
- Kontrolowany dostęp do poszczególnych komponentów

2.7 Diagram architektury



3. Szczegółowy opis komponentów

3.1 MS SQL Server - Baza główna (SchoolDB)

3.1.1 Struktura tabel

Główna baza danych systemu zawiera następujące grupy tabel:

Tabele podstawowe:

- **students**: Dane uczniów (imię, nazwisko, data urodzenia, płeć, grupa)
- **teachers**: Dane nauczycieli (dane osobowe, kontakt, specjalizacja)
- **parents**: Dane rodziców (kontakt, informacje dodatkowe)
- **parents_students**: Relacja rodzic-uczeń (many-to-many)

Tabele organizacyjne:

- **groups**: Grupy klasowe z przypisanym wychowawcą i rokiem szkolnym
- **years**: Lata szkolne
- **classrooms**: Sale lekcyjne z lokalizacją
- **subjects**: Przedmioty nauczania
- **hours**: Godziny lekcyjne (przedziały czasowe)
- **days**: Dni tygodnia

Tabele edukacyjne:

- **marks**: Oceny uczniów (wartość, komentarz, waga, przedmiot)
- **lessons**: Lekcje (nauczyciel, przedmiot, grupa, sala, czas)
- **attendances**: Rejestry obecności na lekcjach
- **attendance_student**: Szczegółowa frekwencja poszczególnych uczniów

Tabele słownikowe:

- **genders**: Płeć (Male, Female, Other)

3.1.2 Kluczowe procedury składowane

Zarządzanie uczniami:

- **sp_CreateStudent**: Dodawanie nowego ucznia z walidacją danych

- **sp_UpdateStudent**: Aktualizacja danych ucznia
- **sp_DeleteStudent**: Usuwanie ucznia z opcją wymuszenia (cascade)
- **sp_GetStudentById**: Pobieranie szczegółowych danych ucznia
- **sp_GetStudentsByGroup**: Lista uczniów w grupie klasowej

Zarządzanie nauczycielami:

- **sp_CreateTeacher**: Rejestracja nowego nauczyciela
- **sp_GetTeacherDetails**: Szczegółowe informacje o nauczycielu i jego obowiązkach

System ocen:

- **sp_AddMark**: Dodawanie oceny z walidacją zakresu (1-6) i wagi
- **sp_GetStudentMarks**: Pobieranie ocen ucznia z przedmiotu
- **sp_GetClassMarksStatistics**: Statystyki ocen dla całej klasy

Frekwencja:

- **sp_GetAttendanceStatistics**: Statystyki frekwencji dla ucznia/grupy

3.1.3 Widoki rozproszone

- **vw_StudentCompleteInfo**: Kompletne informacje o uczniach z danymi rodziców
- **vw_StudentFinancialInfo**: Dane uczniów połączone z informacjami finansowymi z Oracle
- **vw_DistributedStudentData**: Zestawienie danych uczniów z wszystkich systemów

3.2 Oracle Database - System finansowy (FINANCE_DB)

3.2.1 Struktura obiektów

Główne tabele:

- **contracts**: Kontrakty z rodzicami (uczeń, rodzic, kwoty, okres obowiązywania)
- **payments**: Płatności (kontrakt, termin, kwota, status, data wpłaty)

Tabele zdalnych schematów:

- **contracts_remote** (REMOTE_DB1): Kopie kontraktów w schemacie zdalnym
- **payment_summary** (REMOTE_DB2): Podsumowania płatności dla raportowania

3.2.2 Kluczowe procedury i funkcje

Zarządzanie kontraktami:

- **sp_CreateContractWithPayments**: Tworzy kontrakt i automatycznie generuje harmonogram płatności miesięcznych
- **sp_GetContractInfo**: Pobiera szczegółowe informacje o kontrakcie z podsumowaniem płatności

Przetwarzanie płatności:

- **sp_ProcessPayment**: Przetwarza płatność (pełna lub częściowa) z aktualizacją statusu
- **sp_DistributedPaymentProcessing**: Zaawansowane przetwarzanie płatności z aktualizacją schematów zdalnych

Funkcje finansowe:

- **fn_CalculateOutstandingBalance**: Oblicza saldo zaległości dla ucznia

Synchronizacja danych:

- **sp_SyncBetweenSchemas**: Synchronizuje dane między schematami lokalnymi i zdalnymi
- **sp_CrossSchemaFinanceReport**: Generuje raporty finansowe z danych rozproszonych

3.2.3 Pakiet PL/SQL - pkg_DistributedFinance

Zaawansowany pakiet zarządzający operacjami finansowymi w środowisku rozproszonym:

- **fn_GetStudentFinanceData**: Funkcja pipelined zwracająca dane finansowe ucznia
- **sp_DistributedPaymentProcessing**: Przetwarzanie płatności z aktualizacją wielu schematów
- **sp_GenerateFinanceReport**: Generator raportów finansowych (summary, detailed, overdue)

3.2.4 Widoki rozproszone

- **vw_DistributedFinanceData**: Widok łączący dane z głównego schematu i schematów zdalnych
- Obsługuje triggery INSTEAD OF dla symulacji operacji UPDATE/INSERT na danych rozproszonych

3.3 PostgreSQL - System uwag (remarks_main)

3.3.1 Struktura schematu

Schemat remarks_main:

- **remark**: Tabela uwag pedagogicznych

Struktura tabeli remark:

- **id**: Identyfikator uwagi (SERIAL)
- **studentId**: Identyfikator ucznia (INTEGER)
- **teacherId**: Identyfikator nauczyciela (INTEGER)
- **value**: Treść uwagi (TEXT)
- **created_date**: Data utworzenia (TIMESTAMP)

3.3.2 Indeksy i optymalizacja

- **idx_remark_student**: Indeks na studentId dla szybkiego wyszukiwania uwag ucznia
- **idx_remark_teacher**: Indeks na teacherId dla wyszukiwania uwag nauczyciela
- **idx_remark_date**: Indeks na created_date dla raportów okresowych

3.3.3 Zarządzanie dostępem

Użytkownik remarks_user:

- Ograniczone uprawnienia do schematu remarks_main
- Hasło: Remarks123
- Używany przez procedury MS SQL Server do zdalnego dostępu

3.3.4 Integracja z MS SQL Server

Procedury bridge w MS SQL Server:

- **pg_add_remark**: Dodaje uwagę do PostgreSQL przez OPENQUERY
- **pg_delete_remark**: Usuwa uwagę z PostgreSQL przez OPENQUERY

Procedury te używają dynamicznego SQL do komunikacji z PostgreSQL, umożliwiając:

- Dodawanie nowych uwag z MS SQL Server
- Usuwanie uwag na podstawie ID
- Zabezpieczenie przed SQL injection przez escape'owanie apostrofów

3.4 Excel - Eksport danych

3.4.1 Struktura eksportu

System umożliwia eksport danych do plików Excel poprzez procedurę **sp_ExportStudentToExcel**.

Arkusze docelowe:

- **Student$**: Podstawowe dane ucznia
- **ContractsPayments$**: Dane finansowe z Oracle
- **Remarks$**: Uwagi z PostgreSQL

3.4.2 Mechanizm eksportu

Wykorzystuje technologię OPENROWSET z Microsoft.ACE.OLEDB.12.0:

- Dynamiczne generowanie ścieżek do plików Excel
- Automatyczne tworzenie arkuszy z odpowiednimi kolumnami
- Eksport danych z wielu źródeł w jednej operacji

3.4.3 Konfiguracja linked server dla Excel

EXCEL_DATA linked server:

- Provider: Microsoft.ACE.OLEDB.12.0

- Format: Excel 12.0 z nagłówkami (HDR=YES)
- Lokalizacja: C:\excel_exports\

3.5 Integracja komponentów

3.5.1 Przepływ danych między komponentami

Scenariusz 1: Dodanie nowego ucznia z kontraktem

1. MS SQL Server: Utworzenie rekordu ucznia
2. MS SQL Server: Powiązanie z rodzicem
3. Oracle: Automatyczne utworzenie kontraktu finansowego
4. Replikacja: Synchronizacja danych ucznia

Scenariusz 2: Kompletny raport ucznia

1. MS SQL Server: Pobranie danych podstawowych
2. Oracle: Pobranie danych finansowych (OPENQUERY)
3. PostgreSQL: Pobranie uwag (OPENQUERY)
4. Połączenie wszystkich danych w jeden raport

3.5.2 Transakcje rozproszone

System implementuje transakcje rozproszone dla operacji krytycznych:

- **sp_AddStudentWithFinanceContract**: Atomowe dodanie ucznia i kontraktu
- **sp_ProcessStudentPayment**: Przetwarzanie płatności z aktualizacją wielu systemów

Wykorzystuje mechanizm SET XACT_ABORT ON dla automatycznego rollback w przypadku błędów.

4. Replikacja i synchronizacja

4.1 Ogólne założenia replikacji

System implementuje replikację transakcyjną w celu zapewnienia wysokiej dostępności i bezpieczeństwa danych krytycznych. Replikacja obejmuje kluczowe dane uczniów, które są automatycznie synchronizowane między serwerem głównym a serwerem repliki w czasie rzeczywistym.

4.2 Architektura replikacji

4.2.1 Komponenty replikacji

Publisher (Wydawca): Główny serwer MS SQL Server z bazą SchoolDB
Subscriber (Subskrybent): Dodatkowa instancja MS SQL Server (port 1434)
Distributor (Dystrybutor): Ten sam serwer co Publisher
Publication: SchoolDB_StudentsOnly

4.2.2 Typ replikacji

Replikacja transakcyjna ciągła:

- **Typ**: Transactional replication
- **Tryb**: Continuous (ciągły)
- **Synchronizacja**: Automatic
- **Metoda sync**: Concurrent

4.2.3 Konfiguracja dystrybutora

Baza dystrybucyjna: distribution
Lokalizacja danych: C:\MSSQL\Data
Folder roboczy: C:\ReplData
Tryb bezpieczeństwa: Windows Authentication

4.3 Publikacja SchoolDB_StudentsOnly

4.3.1 Artykuły replikacji

Replikowana tabela: students
Typ artykułu: logbased
Zarządzanie tożsamością: MANUAL
Replikacja DDL: Włączona (replicate_ddl = 1)

4.3.2 Właściwości publikacji

- **Częstotliwość replikacji:** Ciągła (continuous)
- **Status publikacji:** Aktywna
- **Tryb aktualizacji subskrybenta:** Read-only
- **Typ subskrypcji:** Push subscription

4.4 Konfiguracja subskrypcji

4.4.1 Parametry subskrypcji

Serwer subskrybenta: 127.0.0.1,1434
Baza docelowa: SchoolDB_Replica
Login subskrybenta: repl_user
Hasło: Pa55w0rd!
Częstotliwość agenta: frequency_type = 64 (ciągły)

4.4.2 Agent dystrybucji

System automatycznie tworzy agenta dystrybucji odpowiedzialnego za:

- Monitorowanie zmian w tabeli students
- Przenoszenie transakcji do bazy dystrybucyjnej
- Aplikowanie zmian na serwerze subskrybenta
- Obsługę konfliktów i błędów replikacji

4.5 Proces inicjalizacji replikacji

4.5.1 Konfiguracja linked servers

Konfiguracja RPC:

- rpc = true
- rpc out = true
- data access = true

Umożliwia zdalne wywoływanie procedur składowanych między serwerami.

4.5.2 Tworzenie snapshot

Agent snapshot:

- Tworzy początkowy obraz danych tabeli students
- Generuje skrypty schema i dane
- Przygotowuje strukturę dla subskrybenta

Wykonanie snapshot:
Automatyczne uruchomienie przez job systemu SQL Server Agent.

4.5.3 Aplikowanie snapshot na subskrybencie

- Utworzenie struktury tabeli students w SchoolDB_Replica
- Załadowanie początkowych danych
- Przygotowanie mechanizmów replikacji ciągłej

5. Linked servers i integracja

5.1 Ogólne założenia linked servers

System wykorzystuje mechanizm linked servers do integracji heterogenicznych systemów bazodanowych. Linked servers umożliwiają wykonywanie zapytań rozproszonych, wywołanie zdalnych procedur składowanych oraz zarządzanie transakcjami rozproszonymi między MS SQL Server, Oracle i PostgreSQL.

5.2 Konfiguracja linked servers

5.2.1 ORACLE_FINANCE - Połączenie z Oracle

Parametry konfiguracji:

- **Nazwa serwera:** ORACLE_FINANCE
- **Provider:** OraOLEDB.Oracle
- **Data Source:** 127.0.0.1:1521/PD19C
- **Product Name:** Oracle

Uwierzytelnianie:

- **Typ:** SQL Server Authentication
- **Remote User:** FINANCE_DB
- **Remote Password:** Finance123

Opcje serwera:

- **RPC:** Enabled (true)
- **RPC Out:** Enabled (true)
- **Data Access:** Enabled (true)

5.2.2 POSTGRES_REMARKS - Połączenie z PostgreSQL

Parametry konfiguracji:

- **Nazwa serwera:** POSTGRES_REMARKS
- **Provider:** MSDASQL (Microsoft OLE DB Provider for ODBC)
- **Product Name:** PostgreSQL
- **Data Source:** PostgreSQL30 (DSN ODBC)

Uwierzytelnianie:

- **Remote User:** remarks_user
- **Remote Password:** Remarks123

Specyfika PostgreSQL:
Wymaga konfiguracji DSN ODBC o nazwie "PostgreSQL30" w systemie operacyjnym.

5.2.3 MSSQL_REPLICA - Replikacja MS SQL Server

Parametry konfiguracji:

- **Nazwa serwera:** MSSQL_REPLICA
- **Provider:** MSOLEDBSQL (Microsoft OLE DB Driver for SQL Server)
- **Data Source:** 127.0.0.1,1434

Uwierzytelnianie:

- **Remote User:** sa
- **Remote Password:** Str0ng!Passw0rd

5.2.4 EXCEL_DATA - Integracja z Excel

Parametry konfiguracji:

- **Nazwa serwera:** EXCEL_DATA
- **Provider:** Microsoft.ACE.OLEDB.12.0
- **Product Name:** Excel
- **Data Source:** C:\excel_exports\SchoolData.xlsx
- **Provider String:** Excel 12.0;HDR=YES;

5.3 Konfiguracja systemowa

5.3.1 Włączenie Ad Hoc Distributed Queries

Wymagane dla operacji OPENROWSET i OPENDATASOURCE:

- **show advanced options:** 1

- **Ad Hoc Distributed Queries:** 1

5.3.2 Uprawnienia i bezpieczeństwo

SQL Server Service Account:
Musi mieć uprawnienia do:

- Łączenia się z serwerami zdalnymi
- Dostępu do plików Excel
- Wykonywania operacji sieciowych

5.4 Typy operacji rozproszonych

5.4.1 OPENQUERY - Zapytania parametryzowane

Zastosowanie: Wykonywanie predefiniowanych zapytań na serwerach zdalnych

Przykład użycia z Oracle:
Pobieranie danych finansowych dla ucznia:

- Zapytanie SQL w składni Oracle
- Automatyczna konwersja typów danych
- Możliwość używania w klauzulach JOIN

Przykład użycia z PostgreSQL:
Pobieranie uwag pedagogicznych:

- Zapytanie SQL w składni PostgreSQL
- Obsługa funkcji agregujących
- Sortowanie i filtrowanie po stronie PostgreSQL

5.4.2 OPENROWSET - Operacje adhoc

Zastosowanie: Bezpośrednie operacje na źródłach zewnętrznych

Excel export/import:

- INSERT do plików Excel
- Dynamiczne tworzenie arkuszy
- Obsługa różnych formatów danych

5.4.3 Remote Procedure Calls (RPC)

Wywoływanie procedur Oracle:
Umożliwia bezpośrednie wywołanie procedur PL/SQL z poziomu T-SQL.

Przekazywanie parametrów:

- INPUT parameters
- OUTPUT parameters
- Complex data types (ograniczone)

5.5 Zapytania rozproszone

5.5.1 Multi-database queries

sp_DistributedStudentReport:
Łączy dane z trzech systemów:

- MS SQL Server: podstawowe dane uczniów i frekwencja
- Oracle: dane finansowe (kontrakty, płatności)
- PostgreSQL: uwagi pedagogiczne

vw_DistributedStudentData:
Widok integrujący dane ze wszystkich źródeł w czasie rzeczywistym.

5.5.2 Transakcje rozproszone

sp_AddStudentWithFinanceContract:
Transakcja obejmująca:

1. Dodanie ucznia w MS SQL Server

2. Utworzenie kontraktu w Oracle
3. Synchronizacja z systemem replik

Mechanizm XACT_ABORT:
Zapewnia atomowość operacji rozproszonych - w przypadku błędu w dowolnym systemie, cała transakcja jest cofana.

5.6 Symulacja linked server - Oracle

5.6.1 Scenariusze testowe

Test podstawowy:
Weryfikacja połączenia i podstawowych operacji SELECT.

Test transakcji:
Sprawdzenie obsługi transakcji rozproszonych z rollback.

Test wydajności:
Benchmarking zapytań rozproszonych vs lokalne operacje.

5.6.2 Przypadki brzegowe

Timeout handling:
Test zachowania systemu przy długo trwających operacjach.

Error propagation:
Weryfikacja propagacji błędów między systemami.

Concurrent access:
Test współbieżnego dostępu do zasobów rozproszonych.

6. Procedury i funkcje

6.1 Klasyfikacja procedur systemu

System zawiera procedury składowane i funkcje podzielone na następujące kategorie funkcjonalne:

- **Procedury CRUD** - podstawowe operacje na danych
- **Procedury raportowania** - generowanie raportów i statystyk
- **Procedury finansowe** - operacje związane z płatnościami i kontraktami
- **Transakcje rozproszone** - operacje obejmujące wiele baz danych
- **Procedury integracyjne** - komunikacja między systemami

6.2 Procedury CRUD - MS SQL Server

6.2.1 Zarządzanie uczniami

sp_CreateStudent

- **Parametry wejściowe:** GroupId, FirstName, LastName, Birthday (opcjonalny), GenderId (opcjonalny)
- **Parametr wyjściowy:** StudentId
- **Funkcjonalność:** Dodaje nowego ucznia z walidacją danych i sprawdzeniem istnienia grupy
- **Walidacje:** Weryfikacja wieku (0-25 lat), sprawdzenie istnienia grupy i płci
- **Transakcje:** Wykorzystuje BEGIN/COMMIT TRANSACTION z obsługą błędów

sp_UpdateStudent

- **Parametry wejściowe:** StudentId, opcjonalne pola do aktualizacji
- **Funkcjonalność:** Aktualizuje dane ucznia używając ISNULL dla zachowania istniejących wartości
- **Walidacje:** Sprawdzenie istnienia ucznia, grupy i płci

sp_DeleteStudent

- **Parametry wejściowe:** StudentId, ForceDelete (bit)
- **Funkcjonalność:** Usuwa ucznia z opcją cascade delete dla powiązanych danych
- **Zabezpieczenia:** Sprawdza istnienie powiązanych ocen i frekwencji

sp_GetStudentById

- **Parametry wejściowe:** StudentId

- **Wynik:** Dwa zestawy wyników - dane ucznia i lista rodziców
- **Funkcjonalność:** Kompletne informacje o uczniu włączając wychowawcę i rok szkolny

sp_GetStudentsByGroup

- **Parametry wejściowe:** GroupId
- **Wynik:** Lista uczniów z podstawowymi statystykami (średnia ocen, liczba ocen)
- **Funkcjonalność:** Agreguje dane ocen dla każdego ucznia w grupie

6.2.2 Zarządzanie nauczycielami

sp_CreateTeacher

- **Parametry wejściowe:** FirstName, LastName, Birthday, PhoneNumber, Email, AdditionalInfo
- **Parametr wyjściowy:** TeacherId
- **Walidacje:** Sprawdzenie formatu email i unikalności
- **Funkcjonalność:** Rejestracja nowego nauczyciela z danymi kontaktowymi

sp_GetTeacherDetails

- **Parametry wejściowe:** TeacherId
- **Wynik:** Trzy zestawy wyników - dane nauczyciela, grupy pod opieką, harmonogram lekcji
- **Funkcjonalność:** Kompleksowy obraz obowiązków nauczyciela

6.2.3 System ocen

sp_AddMark

- **Parametry wejściowe:** SubjectId, StudentId, Value, Comment, Weight
- **Parametr wyjściowy:** MarkId
- **Walidacje:** Zakres oceny (1-6), waga (1-10), istnienie ucznia i przedmiotu
- **Funkcjonalność:** Dodaje ocenę z komentarzem i wagą

sp_GetStudentMarks

- **Parametry wejściowe:** StudentId, SubjectId (opcjonalny)
- **Wynik:** Lista ocen i podsumowanie statystyczne
- **Funkcjonalność:** Szczegółowe oceny oraz średnie ważone i proste

6.2.4 Frekwencja

sp_GetAttendanceStatistics

- **Parametry wejściowe:** StudentId, GroupId, zakres dat
- **Wynik:** Statystyki frekwencji z procentem obecności
- **Funkcjonalność:** Analiza frekwencji dla ucznia lub grupy w określonym okresie

6.3 Procedury raportowania

6.3.1 Raporty akademickie

sp_GetClassMarksStatistics

- **Parametry wejściowe:** GroupId, SubjectId (opcjonalny)
- **Wynik:** Statystyki ocen dla poszczególnych uczniów i całej klasy
- **Funkcjonalność:** Dwupoziomowa analiza - indywidualna i klasowa

6.3.2 Raporty rozproszone

sp_DistributedStudentReport

- **Parametry wejściowe:** StartDate, EndDate
- **Funkcjonalność:** Integruje dane z trzech systemów (MS SQL, Oracle, PostgreSQL)
- **Wynik:** Kompleksowy raport uczniów z danymi finansowymi, uwagami i frekwencją
- **Technologie:** Wykorzystuje OPENQUERY dla dostępu do zdalnych danych

sp_AggregatedReport

- **Funkcjonalność:** Generuje zagregowane statystyki ze wszystkich systemów
- **Wynik:** Trzy sekcje - statystyki uczniów, finansowe i uwag pedagogicznych

- **Zastosowanie:** Raporty zarządcze dla administracji szkoły

6.4 Procedury finansowe - Oracle

6.4.1 Zarządzanie kontraktami

sp_CreateContractWithPayments

- **Parametry wejściowe:** p_student_id, p_parent_id, p_start_date, p_end_date, p_monthly_amount
- **Parametr wyjściowy:** p_contract_id
- **Funkcjonalność:** Tworzy kontrakt i automatycznie generuje harmonogram płatności miesięcznych
- **Logika:** Pętla ADD_MONTHS dla tworzenia płatności od daty rozpoczęcia do zakończenia

sp_GetContractInfo

- **Parametry wejściowe:** p_student_id
- **Wynik:** Kursor z szczegółami kontraktu i podsumowaniem płatności
- **Funkcjonalność:** Kompletne informacje finansowe dla ucznia

6.4.2 Przetwarzanie płatności

sp_ProcessPayment

- **Parametry wejściowe:** p_payment_id, p_paid_amount, p_paid_date
- **Parametr wyjściowy:** p_result (status operacji)
- **Funkcjonalność:** Obsługuje płatności częściowe i pełne
- **Logika:** Sprawdza kwotę i aktualizuje status lub zmniejsza należność

6.4.3 Synchronizacja danych

sp_SyncBetweenSchemas

- **Parametry wejściowe:** p_operation ('SYNC_TO_REMOTE' lub 'SYNC_FROM_REMOTE')
- **Funkcjonalność:** Synchronizuje kontrakty między schematami lokalnymi i zdalnymi
- **Zabezpieczenia:** Sprawdza duplikaty przed insertem

6.5 Pakiet PL/SQL - pkg_DistributedFinance

6.5.1 Funkcje pipelined

fn_GetStudentFinanceData

- **Parametry wejściowe:** p_student_id
- **Zwracany typ:** t_finance_table PIPELINED
- **Funkcjonalność:** Zwraca strukturę danych finansowych ucznia
- **Kalkulacje:** Automatyczne obliczanie salda zaległości

6.5.2 Procedury rozproszone

sp_DistributedPaymentProcessing

- **Parametry wejściowe:** p_student_id, p_payment_amount, p_payment_date
- **Parametr wyjściowy:** p_result
- **Funkcjonalność:** Przetwarza płatność z aktualizacją wielu schematów
- **Integralność:** Wykorzystuje MERGE dla aktualizacji payment_summary

sp_GenerateFinanceReport

- **Parametry wejściowe:** p_report_type ('SUMMARY', 'DETAILED', 'OVERDUE')
- **Parametr wyjściowy:** p_cursor (SYS_REFCURSOR)
- **Funkcjonalność:** Generator różnych typów raportów finansowych

6.6 Procedury integracyjne PostgreSQL

6.6.1 Bridge procedures w MS SQL Server

pg_add_remark

- **Parametry wejściowe:** @studentId, @teacherId, @value

- **Funkcjonalność:** Dodaje uwagę do PostgreSQL przez OPENQUERY
- **Bezpieczeństwo:** Escape'owanie apostrofów dla ochrony przed SQL injection

pg_delete_remark

- **Parametry wejściowe:** @id
- **Funkcjonalność:** Usuwa uwagę z PostgreSQL przez OPENQUERY
- **Implementacja:** Dynamiczny SQL z bezpiecznym przekazywaniem parametrów

6.7 Transakcje rozproszone

6.7.1 Kompleksowe operacje

sp_AddStudentWithFinanceContract

- **Parametry wejściowe:** Dane ucznia, rodzica i kontraktu
- **Parametry wyjściowe:** StudentId, OracleContractId
- **Funkcjonalność:** Atomowa operacja dodania ucznia, powiązania z rodzicem i utworzenia kontraktu
- **Technologie:** SET XACT_ABORT ON, wywołanie zdalnej procedury Oracle
- **Rollback:** Automatyczny w przypadku błędu w dowolnym systemie

sp_ProcessStudentPayment

- **Parametry wejściowe:** StudentId, PaidAmount, PaidDate
- **Funkcjonalność:** Identyfikuje kontrakt, znajduje zaległą płatność, przetwarza wpłatę
- **Integracja:** Wywołuje procedury Oracle przez linked server
- **Walidacje:** Sprawdza istnienie kontraktu i zaległych płatności

6.8 Procedury eksportu

6.8.1 Eksport do Excel

sp_ExportStudentToExcel

- **Parametry wejściowe:** StudentId
- **Funkcjonalność:** Eksportuje dane ucznia do trzech arkuszy Excel
- **Źródła danych:** MS SQL Server (student), Oracle (finance), PostgreSQL (remarks)
- **Technologia:** OPENROWSET z Microsoft.ACE.OLEDB.12.0

6.9 Funkcje pomocnicze Oracle

6.9.1 Kalkulacje finansowe

fn_CalculateOutstandingBalance

- **Parametry wejściowe:** p_student_id
- **Zwracany typ:** NUMBER
- **Funkcjonalność:** Oblicza saldo zaległości na podstawie kontraktów i płatności
- **Kalkulacje:** MONTHS_BETWEEN dla okresu kontraktu, agregacja płatności

6.10 Obsługa błędów i logowanie

MS SQL Server:

- TRY/CATCH blocks we wszystkich procedurahkrytycznych
- RAISERROR z właściwymi poziomami severity
- Logowanie błędów do tabel systemowych

Oracle:

- EXCEPTION handling w blokach PL/SQL
- DBMS_OUTPUT dla debugowania
- Rollback w przypadku błędów krytycznych

Transakcje rozproszone:

- SET XACT_ABORT ON dla automatycznego rollback
- Walidacja wyników zdalnych procedur
- Propagacja błędów między systemami

7. Bezpieczeństwo i zarządzanie dostępem

7.1 Ogólna koncepcja bezpieczeństwa

System implementuje wielowarstwową strategię bezpieczeństwa obejmującą kontrolę dostępu na poziomie bazy danych, szyfrowanie komunikacji, auditing operacji oraz zarządzanie tożsamościami użytkowników w środowisku rozproszonym.

7.2 Architektura bezpieczeństwa

7.2.1 Warstwy zabezpieczeń

Warstwa sieciowa:

- Firewall rules dla portów bazodanowych (1433, 1521, 5432)

Warstwa uwierzytelniania:

- Windows Authentication (preferowane)
- SQL Server Authentication dla aplikacji

Warstwa autoryzacji:

- Role-based access control (RBAC)
- Principle of least privilege
- Granular permissions na poziomie obiektów

Warstwa audingu:

- SQL Server Audit
- Database-level auditing
- Application-level logging

7.3 Zarządzanie użytkownikami - MS SQL Server

7.3.1 Główne role systemu

db_owner (Administratorzy systemu):

- Pełny dostęp do wszystkich obiektów
- Możliwość modyfikacji struktury bazy
- Dostęp do wszystkich procedur administracyjnych
- Zarządzanie linked servers

db_datawriter + db_datareader (Nauczyciele):

- Odczyt wszystkich danych uczniów
- Dodawanie i modyfikacja ocen
- Dostęp do procedur oceniania (sp_AddMark, sp_GetStudentMarks)
- Dostęp do systemu uwag przez procedury bridge

db_datareader (Rodzice - w przyszłości):

- Ograniczony dostęp tylko do danych własnych dzieci
- Widoki filtrujące dane na podstawie parent_id
- Brak dostępu do danych innych uczniów

7.3.2 Procedury systemowe z kontrolą dostępu

sp_CreateStudent, sp_UpdateStudent, sp_DeleteStudent:
Wymagają uprawnień db_datawriter i sprawdzają:

- Poprawność ID użytkownika wykonującego operację
- Uprawnienia do modyfikacji danych grupy
- Logowanie wszystkich operacji modyfikujących

sp_GetCompleteStudentInfo:

7.4 Zarządzanie dostępem - Oracle

7.4.1 Użytkownicy i schematy

FINANCE_DB (Główny schemat finansowy):

- **Hasło:** Finance123
- **Default tablespace:** USERS
- **Quota:** UNLIMITED na USERS
- **Uprawnienia:** CONNECT, RESOURCE, CREATE SEQUENCE, CREATE VIEW

7.4.2 Schematy zdalne

REMOTE_DB1 i REMOTE_DB2:

- Dedykowane użytkownicy z ograniczonymi uprawnieniami
- Kontrolowany dostęp z głównego schematu FINANCE_DB
- Szyfrowanie komunikacji między schematami

7.4.3 Zabezpieczenia PL/SQL

Procedury finansowe:

- Walidacja parametrów wejściowych przeciwko SQL injection
- Sprawdzanie uprawnień na poziomie logiki biznesowej
- Logowanie wszystkich operacji finansowych

Funkcje kalkulacyjne:

- Enkapsulacja logiki w funkcjach bez bezpośredniego dostępu do tabel

7.5 Zarządzanie dostępem - PostgreSQL

7.5.1 Konfiguracja użytkowników

remarks_user:

- **Hasło:** Remarks123
- **Schema:** remarks_main
- **Uprawnienia:** USAGE na schema, SELECT/INSERT/UPDATE/DELETE na tabeli remark
- **Ograniczenia:** Brak możliwości tworzenia obiektów

7.5.2 Row Level Security (RLS)

Przygotowanie do implementacji RLS dla:

- Filtrowanie uwag na podstawie uprawnień nauczyciela
- Ograniczenie dostępu rodziców do uwag własnych dzieci
- Auditing dostępu do wrażliwych uwag

7.6 Linked Servers - Bezpieczeństwo

7.6.1 Mapowanie loginów

ORACLE_FINANCE:

- **Local login:** NULL (wszystkie użytkownicy)
- **Remote login:** FINANCE_DB
- **Security context:** Controlled przez procedury MS SQL Server

POSTGRES_REMARKS:

- **Local login:** NULL
- **Remote login:** remarks_user
- **Connection pooling:** Ograniczone dla kontroli dostępu

8. Wdrożenie i konfiguracja

8.1 Wymagania systemowe

8.1.1 Środowisko MS SQL Server

Wymagane komponenty:

- SQL Server Database Engine
- SQL Server Agent (dla replikacji i job'ów)
- SQL Server Management Studio (SSMS)
- Microsoft OLE DB Driver for SQL Server

8.1.2 Środowisko Oracle

Wymagania:

- Oracle Database 19c lub nowszy
- Oracle Client 19c na serwerze MS SQL
- OraOLEDB.Oracle provider
- TNS configuration dla linked server

Konfiguracja sieciowa:

- Port 1521 otwarty w firewall
- TNS_ADMIN poprawnie skonfigurowane
- ORACLE_HOME w zmiennych środowiskowych

8.1.3 Środowisko PostgreSQL

Wymagania:

- PostgreSQL 13 lub nowszy
- PostgreSQL ODBC Driver (psqlODBC)
- DSN ODBC o nazwie "PostgreSQL30"

Konfiguracja:

- Port 5432 dostępny z serwera MS SQL
- pg_hba.conf skonfigurowane dla połączeń zdalnych
- SSL włączone dla bezpiecznych połączeń

8.2 Kolejność wdrożenia

8.2.1 Faza 1: Przygotowanie środowiska

Krok 1: Instalacja i konfiguracja serwerów

1. Instalacja MS SQL Server
2. Instalacja Oracle Database
3. Instalacja PostgreSQL
4. Konfiguracja sieciowa i firewall rules

Krok 2: Instalacja komponentów integracyjnych

1. Oracle Client na serwerze MS SQL
2. PostgreSQL ODBC Driver na serwerze MS SQL
3. Microsoft.ACE.OLEDB.12.0 dla integracji Excel

8.2.2 Faza 2: Konfiguracja baz danych

Kolejność wykonania skryptów:

MS SQL Server - Baza główna:

```
1. database_setups/main_mssql/create.sql
2. database_setups/main_mssql/data.sql
```

Oracle - System finansowy:

```
1. database_setups/oracle/1_user.sql
2. database_setups/oracle/2_finance_create.sql
3. database_setups/oracle/3_remote_db1.sql
4. database_setups/oracle/4_remote_db2.sql
5. database_setups/oracle/5_finance_synonyms.sql
6. database_setups/oracle/6_finance_data.sql
7. database_setups/oracle/7_remote_db1_data.sql
8. database_setups/oracle/8_remote_db2_data.sql
```

PostgreSQL - System uwag:

```
1. database_setups/postgres/create.sql
2. database_setups/postgres/data.sql
```

8.2.3 Faza 3: Konfiguracja połączeń

Linked Servers (kolejność wykonania):

```
1. linked_servers/config.sql        # Włączenie distributed queries
2. linked_servers/oracle.sql        # Połączenie z Oracle
3. linked_servers/postgres.sql      # Połączenie z PostgreSQL
4. linked_servers/excel.sql         # Połączenie z Excel
5. linked_servers/2msql.sql         # Połączenie replik MS SQL
```

Testowanie połączeń:

```
-- Test Oracle
SELECT * FROM ORACLE_FINANCE..FINANCE_DB.CONTRACTS

-- Test PostgreSQL
SELECT * FROM [POSTGRES_REMARKS].[school].[remarks_main].[remark]

-- Test Excel (wymaga pliku docelowego)
SELECT * FROM EXCEL_DATA...[Sheet1$]
```

8.2.4 Faza 4: Implementacja logiki biznesowej

MS SQL Server - Procedury podstawowe:

```
1. mssql/basic_crud/students/*.sql   # Procedury zarządzania uczniami
2. mssql/basic_crud/teachers/*.sql   # Procedury zarządzania nauczycielami
3. mssql/views/*.sql                 # Widoki systemowe
```

Oracle - Procedury finansowe:

```
1. oracle/finance/procedures/*.sql   # Procedury finansowe
2. oracle/finance/functions/*.sql    # Funkcje kalkulacyjne
3. oracle/finance/package_distributed_finance/head.sql  # Nagłówek pakietu
4. oracle/finance/package_distributed_finance/body.sql  # Implementacja pakietu
5. oracle/finance/views/*.sql        # Widoki finansowe
6. oracle/finance/triggers/*.sql     # Triggery
```

MS SQL Server - Integracja:

```
1. mssql/basic_crud/teachers/postgres/*.sql  # Procedury bridge PostgreSQL
2. mssql/stored_procedures/*.sql             # Procedury rozproszone
3. mssql/transactions.sql/*.sql              # Transakcje rozproszone
4. mssql/excel_export/*.sql                  # Procedury eksportu
```

8.2.5 Faza 5: Konfiguracja replikacji

Przygotowanie replikacji:

```
1. mssql/replikacja.sql  # Konfiguracja dystrybutora i publikacji
```

Kroki konfiguracji replikacji:

1. Konfiguracja dystrybutora na serwerze głównym
2. Utworzenie publikacji SchoolDB_StudentsOnly
3. Dodanie artykułu (tabela students)
4. Konfiguracja subskrybenta na drugim serwerze
5. Utworzenie i uruchomienie snapshot
6. Uruchomienie agentów replikacji

8.3 Konfiguracja szczegółowa

8.3.1 Konfiguracja Oracle Linked Server

TNS Configuration (tnsnames.ora):

```
PD19C =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = 127.0.0.1)(PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = PD19C)
    )
  )
```

Zmienne środowiskowe:

- ORACLE_HOME=C:\oracle\product\19.0.0\client_1
- TNS_ADMIN=C:\oracle\product\19.0.0\client_1\network\admin
- PATH zawiera ORACLE_HOME\bin

8.3.2 Konfiguracja PostgreSQL ODBC

DSN Configuration:

- Data Source Name: PostgreSQL30
- Database: school
- Server: localhost (lub IP PostgreSQL)
- Port: 5432
- User Name: remarks_user
- SSL Mode: require

8.3.3 Konfiguracja Excel Integration

Wymagane foldery:

```
C:\excel_exports\  # Folder dla plików Excel
C:\excel_exports\templates\  # Szablony plików
```

Uprawnienia:

- SQL Server Service Account musi mieć Full Control do C:\excel_exports\
- Folder musi być dostępny z poziomu SQL Server Agent

9. Podsumowanie

9.1 Osiągnięte cele projektu

System zarządzania bazą danych prywatnej szkoły podstawowej został pomyślnie zrealizowany zgodnie z założeniami projektowymi. Wszystkie główne cele zostały osiągnięte:

9.1.1 Implementacja architektury rozproszonej

- **Heterogeniczna integracja**: Pomyślnie zintegrowano trzy różne systemy bazodanowe (MS SQL Server, Oracle, PostgreSQL)
- **Modularność funkcjonalna**: Każdy komponent odpowiada za określony obszar biznesowy (edukacyjny, finansowy, uwagi)
- **Skalowalność**: Architektura umożliwia łatwe dodawanie nowych komponentów i funkcjonalności

9.1.2 Funkcjonalność biznesowa

- **Zarządzanie uczniami i nauczycielami**: Kompletny system CRUD z walidacją danych
- **System ocen i frekwencji**: Zaawansowane mechanizmy raportowania i statystyk
- **Moduł finansowy**: Automatyzacja kontraktów, płatności i rozliczeń
- **System uwag pedagogicznych**: Narzędzie komunikacji między nauczycielami a administracją

9.1.3 Zaawansowane funkcje techniczne

- **Replikacja transakcyjna**: Zapewnienie ciągłości działania poprzez automatyczną synchronizację danych
- **Linked servers**: Umożliwienie zapytań rozproszonych między heterogenicznymi systemami
- **Transakcje rozproszone**: Gwarancja atomowości operacji obejmujących wiele baz danych
- **Eksport danych**: Integracja z systemami zewnętrznymi (Excel) dla raportowania

9.2 Kluczowe osiągnięcia techniczne

9.2.1 Integracja systemów

System demonstruje zaawansowane techniki integracji:

- **27 procedur składowanych** realizujących kompleksową logikę biznesową
- **4 linked servers** umożliwiających komunikację między systemami
- **Pakiet PL/SQL** z funkcjami pipelined dla zaawansowanych operacji finansowych
- **Widoki rozproszone** integrujące dane z wszystkich źródeł

9.2.2 Bezpieczeństwo i niezawodność

- **Wielowarstwowa kontrola dostępu** z rolami i uprawnieniami na każdym poziomie
- **Replikacja transakcyjna** zapewniająca wysoką dostępność danych krytycznych
- **Auditing i monitoring** wszystkich operacji modyfikujących dane
- **Transakcje rozproszone** z automatycznym rollback w przypadku błędów

9.2.3 Wydajność i optymalizacja

- **Indeksy** optymalizujące najczęściej wykonywane zapytania
- **Procedury parametryzowane** zapobiegające SQL injection
- **Optymalizacja zapytań rozproszonych** minimalizująca transfer danych
- **Cache'owanie połączeń** linked servers dla lepszej wydajności

9.3 Wnioski techniczne

9.3.1 Skuteczność linked servers

Mechanizm linked servers okazał się bardzo skuteczny dla:

- Wykonywania zapytań rozproszonych w czasie rzeczywistym
- Wywołania zdalnych procedur składowanych
- Zarządzania transakcjami rozproszonymi

9.3.2 Korzyści z heterogeniczności

Wykorzystanie różnych systemów bazodanowych przyniosło korzyści:

- **Oracle**: Doskonale sprawdził się w operacjach finansowych wymagających precyzji
- **PostgreSQL**: Efektywny dla prostych operacji CRUD na uwagach
- **MS SQL Server**: Idealny jako centralny hub integrujący wszystkie systemy

9.3.3 Wyzwania i rozwiązania

Główne wyzwania i sposoby ich rozwiązania:

- **Mapowanie typów danych**: Rozwiązane przez staranne projektowanie interfejsów
- **Zarządzanie połączeniami**: Optymalizacja przez connection pooling i timeouts
- **Synchronizacja transakcji**: Użycie XACT_ABORT ON dla atomowości

9.4 Możliwości rozwoju

9.4.1 Krótkoterminowe usprawnienia

- **Interface użytkownika**: Rozwój aplikacji webowej lub desktopowej
- **Mobile app**: Aplikacja mobilna dla rodziców i nauczycieli
- **Notyfikacje**: System alertów email/SMS dla ważnych wydarzeń
- **Backup automation**: Automatyzacja procedur backup i recovery

9.4.2 Długoterminowe rozszerzenia

- **Analityka predykcyjna**: Machine learning dla przewidywania wyników uczniów
- **CRM rodziców**: Rozszerzony system komunikacji z rodzicami
- **Integracja bankowa**: Automatyczne dopasowywanie wpłat
- **Cloud migration**: Migracja do architektury chmurowej

9.4.3 Skalowanie systemu

- **Multi-tenant**: Obsługa wielu szkół w jednym systemie
- **Geograficzne rozposzenie**: Replikacja między regionami
- **Microservices**: Ewolucja w kierunku architektury mikroserwisów