

Politechnika Łódzka

Wydział Elektrotechniki, Elektroniki, Informatyki i Automatyki

Szkoła Muzyczna I Stopnia

Obiektowa Baza Danych Oracle

Rozproszone i Obiektowe Bazy Danych

Autorzy: Igor Typiński (251237)
Mateusz Mróz (251190)

Grupa: 5

Temat: Szkoła muzyczna (z naciskiem na rozwój ucznia)

Łódź, luty 2026

Spis treści

1	Opis projektu	3
1.1	Cel i zakres	3
1.2	Przyjęte ograniczenia	3
1.3	Technologia	4
2	Typy obiektowe	5
2.1	t_wyposazenie	5
2.2	t_przedmiot	5
2.3	t_grupa	5
2.4	t_nauczyciel	6
2.5	t_sala	6
2.6	t_uczen	6
2.7	t_lekcja	6
2.8	t_ocena	7
3	Tabele obiektowe	8
3.1	przedmioty	8
3.2	grupy	8
3.3	nauczyciele	8
3.4	sale	8
3.5	uczniowie	9
3.6	lekcje	9
3.7	oceny	9
3.8	Referencje (REF/DEREF)	9
3.9	Sekwencje	9
4	Pakiety PL/SQL	11
4.1	pkg_slovniki	11
4.2	pkg_osoby	11
4.3	pkg_lekcje	12
4.4	pkg_oceny	12
4.5	pkg_raporty	12
5	Wyzwalacze (Triggery)	14
5.1	trg_lekcja_xor	14
5.2	trg_ocena_zakres	14
6	Obsługa błędów	15
6.1	Kody błędów aplikacji	15
6.2	Przykład komunikatu z sugestią	15
7	Heurystyka sugestii terminu (First Fit)	16
7.1	Algorytm znajdz_alternatywe()	16
7.2	Dopasowanie sali	16
7.3	Funkcja sala_ma_instrument()	16

8	Scenariusze użycia	18
8.1	Scenariusz 1: Administrator rozszerza ofertę szkoły	18
8.2	Scenariusz 2: Sekretariat tworzy grupę i zapisuje uczniów	18
8.3	Scenariusz 3: Planowanie lekcji grupowych	18
8.4	Scenariusz 4: Planowanie lekcji indywidualnych z konfliktem	18
8.5	Scenariusz 5: Nauczyciel wystawia oceny	18
8.6	Scenariusz 6: Uczeń sprawdza oceny i plan	18
8.7	Scenariusz 7: Raporty dla dyrekcji	19
8.8	Scenariusz 8: Walidacje i przypadki brzegowe	19
8.9	Scenariusz 9: Demonstracja VARRAY	19
8.10	Scenariusz 10: Demonstracja REF/DEREF	19
8.11	Scenariusz 11: Demonstracja metod obiektowych	19
9	Diagram relacji obiektów	20
10	Struktura plików projektu	21
11	Podsumowanie wymagań	22

1 Opis projektu

1.1 Cel i zakres

Projekt przedstawia obiektową bazę danych dla szkoły muzycznej I stopnia, ze szczególnym uwzględnieniem śledzenia rozwoju uczniów. Szkoła prowadzi nauczanie gry na instrumentach w trybie indywidualnym oraz zajęcia grupowe (kształcenie słuchu, rytmika). System został zaprojektowany z myślą o codziennej pracy sekretariatu, nauczycieli oraz dyrekcji szkoły.

Głównym celem projektu jest stworzenie kompleksowego systemu informatycznego umożliwiającego:

- **Zarządzanie danymi uczniów** – rejestracja nowych uczniów, przechowywanie danych osobowych, automatyczne obliczanie wieku, przypisanie do grupy (klasy) przez referencję REF
- **Zarządzanie danymi nauczycieli** – ewidencja kadry pedagogicznej, każdy nauczyciel uczy jednego przedmiotu
- **Planowanie lekcji** – tworzenie harmonogramu zajęć indywidualnych (1 uczeń) i grupowych (cała klasa) z walidacją konfliktów terminów oraz heurystyką sugestii wolnego terminu
- **Rezerwacja sal lekcyjnych** – przydzielanie sal z odpowiednim wyposażeniem (przechowywane jako VARRAY) do poszczególnych lekcji
- **Ocenianie postępu uczniów** – wystawianie ocen częściowych i semestralnych z walidacją uprawnień nauczyciela
- **Kontrola reguł biznesowych** – automatyczna walidacja uczestników lekcji (albo uczeń albo grupa), zakresu ocen, konfliktów czasowych, kompetencji nauczycieli
- **Generowanie raportów** – plan dnia, plan ucznia, plan nauczyciela, raport grup, statystyki szkoły

System uwzględnia specyfikę szkoły muzycznej I stopnia z 6-letnim cyklem nauczania. Lekcje odbywają się w godzinach popołudniowych (14:00–20:00), co odpowiada potrzebom uczniów uczęszczających równoległe do szkoły ogólnokształcącej.

1.2 Przyjęte ograniczenia

W projekcie przyjęto następujące ograniczenia biznesowe:

1. **Cykl nauczania:** 6 lat – klasy I-VI szkoły muzycznej I stopnia
2. **Godziny pracy szkoły:** 14:00–20:00 – lekcje mogą być planowane tylko w tych godzinach
3. **Dni pracy szkoły:** poniedziałek–piątek – szkoła nie prowadzi zajęć w weekendy
4. **Czas trwania lekcji:** stały 45 minut dla wszystkich zajęć

5. **Siatka godzinowa:** lekcje rozpoczynają się o pełnych godzinach (14:00, 15:00, 16:00...)
6. **Jeden instrument na ucznia:** każdy uczeń uczy się jednego instrumentu głównego
7. **Jeden przedmiot na nauczyciela:** każdy nauczyciel uczy jednego przedmiotu
8. **XOR lekcji:** lekcja jest ALBO indywidualna (1 uczeń) ALBO grupowa (klasa) – nigdy obie opcje jednocześnie
9. **Maksymalne wyposażenie sali:** 10 elementów (ograniczenie VARRAY)
10. **Skala ocen:** 1–6 (polska skala szkolna)
11. **Typy sal:** indywidualna lub grupowa
12. **Typy przedmiotów:** indywidualny (instrument) lub grupowy (kształcenie słuchu, rytmika)
13. **Brak konfliktów czasowych:** ta sama sala, nauczyciel, uczeń lub grupa nie mogą mieć dwóch lekcji w tym samym czasie
14. **Walidacja kompetencji:** nauczyciel może prowadzić tylko lekcje z przedmiotu, który jest do niego przypisany
15. **Walidacja typu sali:** lekcje grupowe wymagają sali typu 'grupowa'

1.3 Technologia

- **Oracle Database** – obiektowo-relacyjna baza danych
- **Podejście obiektowo-relacyjne** – typy obiektowe z metodami, REF/DEREF, VARRAY
- **Język PL/SQL** – pakiety, procedury, funkcje, trigger

2 Typy obiektowe

W projekcie zdefiniowano 8 typów obiektowych z łącznie 10 metodami.

Typ	Metody	Opis
t_wyposazenie	–	VARRAY(10) nazw elementów wyposażenia
t_przedmiot	1	Przedmiot nauczania (indywidualny/grupowy)
t_grupa	–	Klasa uczniów (symbol, poziom I-VI)
t_nauczyciel	2	Nauczyciel z REF do przedmiotu
t_sala	2	Sala lekcyjna z VARRAY wyposażenia
t_uczen	2	Uczeń z REF do grupy
t_lekcja	2	Lekcja z wyborem: REF do ucznia lub grupy
t_ocena	1	Ocena postępu ucznia
Razem:		8 typów, 10 metod

Tabela 1: Typy obiektowe w projekcie

2.1 t_wyposazenie

Kolekcja VARRAY przechowująca nazwy elementów wyposażenia sali lekcyjnej.

```
CREATE OR REPLACE TYPE t_wyposazenie AS VARRAY(10) OF VARCHAR2(50);
```

Ograniczenie do 10 elementów wynika z założenia, że sala ma skończoną liczbę istotnych elementów wyposażenia (fortepian, metronom, lustro, pulpity itp.). Kolekcja przeszukiwana jest przez funkcję prywatną `sala_ma_instrument()` przy planowaniu lekcji.

2.2 t_przedmiot

Reprezentuje przedmiot nauczania w słowniku przedmiotów.

Atrybuty:

- id (NUMBER) – unikalny identyfikator
- nazwa (VARCHAR2(50)) – nazwa przedmiotu (np. Fortepian, Kształcenie słuchu)
- typ (VARCHAR2(20)) – typ: 'indywidualny' lub 'grupowy'
- czas_min (NUMBER) – czas trwania lekcji (stałe 45 minut)

Metody:

- czy_grupowy() RETURN VARCHAR2 – zwraca 'T' jeśli typ = 'grupowy', inaczej 'N'

2.3 t_grupa

Reprezentuje klasę (grupę) uczniów w szkole muzycznej.

Atrybuty:

- id (NUMBER) – unikalny identyfikator
- symbol (VARCHAR2(10)) – symbol grupy (np. '1A', '2A', '3A')
- poziom (NUMBER) – poziom klasy 1-6 (klasa I-VI szkoły muzycznej)

Typ nie posiada metod – jest prostym kontenerem danych słownikowych.

2.4 t_nauczyciel

Reprezentuje nauczyciela szkoły muzycznej wraz z referencją do przedmiotu.

Atrybuty:

- id (NUMBER) – unikalny identyfikator
- imie (VARCHAR2(50)) – imię nauczyciela
- nazwisko (VARCHAR2(50)) – nazwisko nauczyciela
- data_zatr (DATE) – data zatrudnienia
- ref_przedmiot (REF t_przedmiot) – referencja do przedmiotu który uczy

Metody:

- pelne_nazwisko() RETURN VARCHAR2 – zwraca imię i nazwisko
- staz_lat() RETURN NUMBER – oblicza liczbę lat pracy w szkole

2.5 t_sala

Reprezentuje salę lekcyjną z kolekcją VARRAY wyposażenia.

Atrybuty:

- id (NUMBER) – unikalny identyfikator
- numer (VARCHAR2(10)) – numer sali (np. '101', '102')
- typ (VARCHAR2(20)) – typ: 'indywidualna' lub 'grupowa'
- pojemnosc (NUMBER) – maksymalna liczba osób
- wyposazenie (t_wyposazenie) – VARRAY elementów wyposażenia

Metody:

- czy_grupowa() RETURN VARCHAR2 – zwraca 'T' jeśli typ = 'grupowa', inaczej 'N'
- lista_wyposazenia() RETURN VARCHAR2 – zwraca wyposażenie jako tekst rozdzielony przecinkami (iteracja po VARRAY)

2.6 t_uczen

Reprezentuje ucznia szkoły muzycznej z referencją do grupy.

Atrybuty:

- id (NUMBER) – unikalny identyfikator
- imie (VARCHAR2(50)) – imię ucznia
- nazwisko (VARCHAR2(50)) – nazwisko ucznia
- data_ur (DATE) – data urodzenia
- instrument (VARCHAR2(50)) – instrument główny (np. Fortepian, Skrzypce)
- ref_grupa (REF t_grupa) – referencja do grupy (klasy)

Metody:

- pelne_nazwisko() RETURN VARCHAR2 – zwraca imię i nazwisko
- wiek() RETURN NUMBER – oblicza aktualny wiek w latach

2.7 t_lekcja

Reprezentuje pojedynczą lekcję muzyki z wyborem uczestnika (albo uczeń albo grupa).

Atrybuty:

- id (NUMBER) – unikalny identyfikator
- ref_przedmiot (REF t_przedmiot) – referencja do przedmiotu
- ref_nauczyciel (REF t_nauczyciel) – referencja do nauczyciela

- ref_sala (REF t_sala) – referencja do sali
- ref_uczen (REF t_uczen) – referencja do ucznia (lekcja indywidualna)
- ref_grupa (REF t_grupa) – referencja do grupy (lekcja grupowa)
- data_lekcji (DATE) – data lekcji
- godz_rozp (NUMBER) – godzina rozpoczęcia (14, 15, 16... – pełne godziny)
- czas_min (NUMBER) – czas trwania w minutach (stałe 45)

Metody:

- godzina_koniec() RETURN NUMBER – oblicza godzinę zakończenia (np. 14.75 dla lekcji 14:00-14:45)
- czy_indywidualna() RETURN VARCHAR2 – zwraca 'T' jeśli ref_uczen IS NOT NULL, inaczej 'N'

Reguła przypisania: Lekcja musi mieć wypełnione ALBO ref_uczen (lekcja indywidualna) ALBO ref_grupa (lekcja grupowa), nigdy oba jednocześnie i nigdy żadne. Walidowane przez trigger trg_lekcja_xor.

2.8 t_ocena

Reprezentuje ocenę postępu ucznia.

Atrybuty:

- id (NUMBER) – unikalny identyfikator
- ref_uczen (REF t_uczen) – referencja do ucznia
- ref_nauczyciel (REF t_nauczyciel) – referencja do nauczyciela wystawiającego
- ref_przedmiot (REF t_przedmiot) – referencja do przedmiotu
- wartosc (NUMBER) – wartość oceny 1–6
- data_oceny (DATE) – data wystawienia oceny
- semestralna (VARCHAR2(1)) – 'T' dla oceny semestralnej, 'N' dla częściowej

Metody:

- opis_oceny() RETURN VARCHAR2 – zwraca ocenę słownie (celujący, bardzo dobry, dobry, dostateczny, dopuszczający, niedostateczny)

3 Tabele obiektowe

Utworzono 7 tabel obiektowych przechowujących dane.

Tabela	Typ bazowy	Referencje / Kolekcje
przedmioty	t_przedmiot	–
grupy	t_grupa	–
nauczyciele	t_nauczyciel	ref_przedmiot
sale	t_sala	zawiera VARRAY (wyposazenie)
uczniowie	t_uczen	ref_grupa
lekcje	t_lekcja	ref_przedmiot, ref_nauczyciel, ref_sala, ref_uczen (indyw.), ref_grupa (grup.)
oceny	t_ocena	ref_uczen, ref_nauczyciel, ref_przedmiot

Tabela 2: Tabele obiektowe i ich referencje

3.1 przedmioty

Słownik przedmiotów nauczania.

- Klucz główny: id
- Ograniczenie UNIQUE: nazwa
- Ograniczenia NOT NULL: nazwa, typ, czas_min
- CHECK: typ IN ('indywidualny', 'grupowy')
- CHECK: czas_min = 45

3.2 grupy

Słownik grup (klas) uczniów.

- Klucz główny: id
- Ograniczenie UNIQUE: symbol
- Ograniczenia NOT NULL: symbol, poziom
- CHECK: poziom BETWEEN 1 AND 6

3.3 nauczyciele

Dane nauczycieli z referencją do przedmiotu.

- Klucz główny: id
- Ograniczenia NOT NULL: imie, nazwisko, data_zatr
- Referencja: ref_przedmiot → przedmioty

3.4 sale

Informacje o salach lekcyjnych z kolekcją VARRAY wyposażenia.

- Klucz główny: id
- Ograniczenie UNIQUE: numer
- Ograniczenia NOT NULL: numer, typ, pojemnosc
- CHECK: typ IN ('indywidualna', 'grupowa')
- CHECK: pojemnosc > 0
- Zawiera kolekcję t_wyposazenie (VARRAY)

3.5 uczniowie

Dane uczniów szkoły muzycznej z referencją do grupy.

- Klucz główny: id
- Ograniczenia NOT NULL: imie, nazwisko, data_ur, instrument
- Referencja: ref_grupa → grupy

3.6 lekcje

Zaplanowane lekcje muzyki z wyborem typu uczestnictwa.

- Klucz główny: id
- Ograniczenia NOT NULL: data_lekcji, godz_rozp, czas_min
- CHECK: godz_rozp BETWEEN 14 AND 19
- CHECK: czas_min = 45
- Referencje: ref_przedmiot, ref_nauczyciel, ref_sala
- Relacja uczestników: ref_uczen (dla indywidualnych), ref_grupa (dla grupowych)
- Trigger: trg_lekcja_xor wymusza poprawność danych

3.7 oceny

Oceny postępu uczniów.

- Klucz główny: id
- Ograniczenia NOT NULL: wartosc, data_oceny, semestralna
- CHECK: wartosc BETWEEN 1 AND 6
- CHECK: semestralna IN ('T', 'N')
- Referencje: ref_uczen, ref_nauczyciel, ref_przedmiot

3.8 Referencje (REF/DEREF)

W projekcie zastosowano 9 referencji do modelowania relacji między obiektami:

- Nauczyciel wskazuje na przedmiot, którego uczy (1 REF)
- Uczeń wskazuje na grupę, do której należy (1 REF)
- Lekcja wskazuje na przedmiot, nauczyciela, salę oraz ucznia lub grupę (4-5 REF)
- Ocena wskazuje na ucznia, nauczyciela i przedmiot (3 REF)

Dzięki Deref możliwe jest odwołanie się do atrybutów i metod obiektu wskazywanego:

```
SELECT Deref(1.ref_uczen).pelne_nazwisko() AS uczen,  
       Deref(1.ref_sala).numer AS sala,  
       Deref(1.ref_przedmiot).nazwa AS przedmiot  
FROM lekcje 1 WHERE 1.ref_uczen IS NOT NULL;
```

3.9 Sekwencje

Utworzono 7 sekwencji do generowania identyfikatorów:

- seq_przedmioty – dla tabeli przedmioty
- seq_grupy – dla tabeli grupy
- seq_nauczyciele – dla tabeli nauczyciele
- seq_sale – dla tabeli sale
- seq_uczniowie – dla tabeli uczniowie

- seq_lekcje – dla tabeli lekcje
- seq_oceny – dla tabeli oceny

4 Pakiety PL/SQL

Logika biznesowa zaimplementowana w 5 pakietach z łącznie 25 podprogramami.

Pakiet	Podprogramy	Funkcjonalności
pkg_slowniki	9	Zarządzanie słownikami (przedmioty, grupy, sale)
pkg_osoby	7	Zarządzanie nauczycielami i uczniami
pkg_lekcje	5 (+3 prywatne)	Planowanie lekcji z walidacją i heurystyką
pkg_oceny	4 (+1 prywatna)	Zarządzanie ocenami
pkg_raporty	2	Raporty i statystyki
Razem:		25 podprogramów publicznych

Tabela 3: Pakiety PL/SQL

4.1 pkg_slowniki

Pakiet do zarządzania słownikami (przedmioty, grupy, sale).

Procedury dodawania:

- `dodaj_przedmiot(p_nazwa, p_typ)` – dodaje przedmiot do słownika
- `dodaj_grupe(p_symbol, p_poziom)` – dodaje grupę (klasę)
- `dodaj_sale(p_numer, p_typ, p_pojemnosc, p_wyposazenie)` – dodaje salę z VAR-RAY wyposażenia

Funkcje pobierania referencji:

- `get_ref_przedmiot(p_id)` RETURN REF t_przedmiot
- `get_ref_grupa(p_id)` RETURN REF t_grupa
- `get_ref_sala(p_id)` RETURN REF t_sala

Procedury listowania:

- `lista_przedmiotow()` – wyświetla listę przedmiotów
- `lista_grup()` – wyświetla listę grup
- `lista_sal()` – wyświetla listę sal z wyposażeniem (wywołuje metodę `lista_wyposazenia()`)

4.2 pkg_osoby

Pakiet do zarządzania nauczycielami i uczniami.

Procedury dodawania:

- `dodaj_nauczyciela(p_imie, p_nazwisko, p_id_przedmiotu)` – dodaje nauczyciela z REF do przedmiotu
- `dodaj_ucznia(p_imie, p_nazwisko, p_data_ur, p_instrument, p_id_grupy)` – dodaje ucznia z REF do grupy

Funkcje pobierania referencji:

- `get_ref_nauczyciel(p_id)` RETURN REF t_nauczyciel
- `get_ref_uczen(p_id)` RETURN REF t_uczen

Procedury listowania:

- `lista_nauczycieli()` – wyświetla nauczycieli z przedmiotami (DEREF)
- `lista_uczniow()` – wyświetla uczniów z grupami (DEREF)

- lista_uczniow_grupy(p_id_grupy) – wyświetla uczniów danej grupy (**kursor jawny** OPEN/FETCH/CLOSE)

4.3 pkg_lekcje

Pakiet do zarządzania lekcjami z pełną walidacją konfliktów i heurystyką sugestii terminu.

Procedury publiczne:

- dodaj_lekcje_indywidualna(...) – planuje lekcję indywidualną z walidacją
- dodaj_lekcje_grupowa(...) – planuje lekcję grupową z walidacją
- plan_ucznia(p_id_ucznia) – wyświetla plan ucznia (UNION lekcji indywidualnych i grupowych)
- plan_nauczyciela(p_id_nauczyciela) – wyświetla plan nauczyciela
- plan_dnia(p_data) – wyświetla wszystkie lekcje w danym dniu

Funkcje prywatne:

- sprawdz_kolizje(...) – sprawdza dostępność sali, nauczyciela, ucznia/grupy
- sala_ma_instrument(p_id_sali, p_instrument) – przeszukuje **VARRAY** wyposażenia sali
- znajdz_alternatywe(...) – **heurystyka First Fit** szukająca wolnego terminu

Walidacje w procedurze dodaj_lekcje_indywidualna:

- Kompetencje nauczyciela (czy uczy tego przedmiotu – sprawdzenie REF)
- Zgodność instrumentu ucznia z przedmiotem
- Konflikt sali – brak nakładających się terminów
- Konflikt nauczyciela – brak nakładających się terminów
- Konflikt ucznia – brak nakładających się terminów

Dodatkowe walidacje w procedurze dodaj_lekcje_grupowa:

- Typ sali – lekcja grupowa wymaga sali typu 'grupowa'
- Przepełnienie sali – pojemność \geq liczba uczniów w grupie
- Konflikt grupy – brak nakładających się terminów

4.4 pkg_oceny

Pakiet do zarządzania ocenami postępu uczniów.

Procedury publiczne:

- wystaw_ocene(p_id_ucznia, p_id_nauczyciela, p_id_przedmiotu, p_wartosc) – ocena częściowa
- wystaw_ocene_semestralna(...) – ocena semestralna (semestralna='T')
- oceny_ucznia(p_id_ucznia) – wyświetla wszystkie oceny ucznia z opisem słownym

Funkcja publiczna:

- srednia_ucznia(p_id_ucznia, p_id_przedmiotu) RETURN NUMBER – średnia ocen częściowych (0 gdy brak)

Procedura prywatna:

- sprawdz_uprawnienia_oceniania(...) – waliduje czy nauczyciel uczy danego przedmiotu

4.5 pkg_raporty

Pakiet do generowania raportów i statystyk.

Procedury:

- `raport_grup()` – wyświetla liczbę uczniów w każdej grupie (podzapytanie skorelowane)
- `statystyki()` – wyświetla ogólne statystyki szkoły (liczba uczniów, nauczycieli, grup, sal, lekcji, ocen)

5 Wyzwalacze (Triggery)

Zdefiniowano 2 wyzwalacze realizujące krytyczne reguły biznesowe. Walidacja konfliktów terminów jest zaimplementowana w pakiecie `pkg_lekcje`, aby uniknąć błędu ORA-04091 (Mutating Table).

Trigger	Typ	Działanie
trg_lekcja_xor	BEFORE INSERT/UPDATE	Weryfikacja uczestników: lekcja musi mieć ALBO ucznia ALBO grupę (nigdy oba, nigdy żadne)
trg_ocena_zakres	BEFORE INSERT/UPDATE	Walidacja zakresu ocen 1-6 z przyjaznym komunikatem błędu

Tabela 4: Wyzwalacze w projekcie

5.1 trg_lekcja_xor

Trigger wymuszający poprawność przypisania uczestników dla lekcji:

```
CREATE OR REPLACE TRIGGER trg_lekcja_xor
BEFORE INSERT OR UPDATE ON lekcje
FOR EACH ROW
BEGIN
    IF (:NEW.ref_uczen IS NULL AND :NEW.ref_grupa IS NULL) OR
        (:NEW.ref_uczen IS NOT NULL AND :NEW.ref_grupa IS NOT NULL) THEN
        RAISE_APPLICATION_ERROR(-20001,
            'Lecja musi miec ALBO ucznia (indywidualna) ALBO grupe (grupowa)');
    END IF;
END;
```

5.2 trg_ocena_zakres

Trigger walidujący zakres ocen z przyjaznym komunikatem:

```
CREATE OR REPLACE TRIGGER trg_ocena_zakres
BEFORE INSERT OR UPDATE ON oceny
FOR EACH ROW
BEGIN
    IF :NEW.wartosc < 1 OR :NEW.wartosc > 6 THEN
        RAISE_APPLICATION_ERROR(-20002,
            'Ocena musi byc w zakresie 1-6. Podano: ' || :NEW.wartosc);
    END IF;
END;
```

6 Obsługa błędów

W projekcie zastosowano mechanizmy obsługi wyjątków z własnymi kodami błędów.

6.1 Kody błędów aplikacji

Kod	Źródło	Znaczenie
-20001	trg_lekcja_xor	Nieprawidłowe przypisanie uczestników (lekcja musi mieć ucznia LUB grupę)
-20002	trg_ocena_zakres	Ocena poza zakresem 1-6
-20010	pkg_slovniki.get_ref_przedmiot	Nie znaleziono przedmiotu o podanym ID
-20011	pkg_slovniki.get_ref_grupa	Nie znaleziono grupy o podanym ID
-20012	pkg_slovniki.get_ref_sala	Nie znaleziono sali o podanym ID
-20013	pkg_osoby.get_ref_nauczyciel	Nie znaleziono nauczyciela o podanym ID
-20014	pkg_osoby.get_ref_uczen	Nie znaleziono ucznia o podanym ID
-20020	pkg_lekcje.dodaj_lekcje_indywidualna	Konflikt terminów (+ sugestia alternatywy)
-20021	pkg_lekcje.dodaj_lekcje_grupowa	Konflikt terminów (+ sugestia alternatywy)
-20030	pkg_lekcje	Nauczyciel nie uczy podanego przedmiotu
-20031	pkg_lekcje	Lekcja grupowa w sali indywidualnej
-20032	pkg_lekcje	Instrument ucznia niezgodny z przedmiotem
-20033	pkg_oceny	Nauczyciel nie ma uprawnień do oceniania
-20035	pkg_lekcje	Przepelnienie sali (grupa > pojemność)

Tabela 5: Kody błędów aplikacji

6.2 Przykład komunikatu z sugestią

Gdy system wykryje konflikt terminów, automatycznie sugeruje alternatywny termin:

ORA-20020: Bład planowania: Sala jest juz zajeta w tym terminie!

SUGEROWANY TERMIN: 2025-06-02 o godzinie 15:00 w sali 101

7 Heurystyka sugestii terminu (First Fit)

System implementuje algorytm **First Fit** do automatycznego sugerowania wolnego terminu w przypadku konfliktu.

7.1 Algorytm `znajdz__alternatywe()`

1. Zaczynij od następnej godziny po nieudanym terminie
2. Sprawdzaj godziny robocze (14:00 – 19:00)
3. Jeśli dzień się skończy, przeskocz do następnego dnia na 14:00
4. Szukaj maksymalnie przez 7 dni (limit bezpieczeństwa)
5. Dla każdego terminu iteruj po dostępnych salach

7.2 Dopasowanie sali

Typ lekcji	Kryteria doboru sali
Indywidualna	Funkcja <code>sala_ma_instrument()</code> przeszukuje VARRAY wyposażenia sali, szukając elementu pasującego do instrumentu ucznia. Obsługuje synonimy (np. Pianino = Fortepian).
Grupowa	Szuka tylko sal typu 'grupowa' z pojemnością \geq liczba uczniów w grupie

Tabela 6: Kryteria doboru sali w heurystyce

7.3 Funkcja `sala_ma_instrument()`

Funkcja przeszukuje kolekcję VARRAY wyposażenia sali:

```
FUNCTION sala_ma_instrument(p_id_sali NUMBER, p_instrument VARCHAR2)
RETURN BOOLEAN IS
    v_wyposazenie t_wyposazenie;
BEGIN
    SELECT s.wyposazenie INTO v_wyposazenie FROM sale s WHERE s.id = p_id_sali;
    IF v_wyposazenie IS NULL THEN RETURN FALSE; END IF;

    FOR i IN 1..v_wyposazenie.COUNT LOOP
        IF UPPER(v_wyposazenie(i)) LIKE '%' || UPPER(p_instrument) || '%' THEN
            RETURN TRUE;
        END IF;
        -- Synonim: Pianino = Fortepian
        IF UPPER(p_instrument) = 'FORTEPIAN' AND
            UPPER(v_wyposazenie(i)) LIKE '%PIANINO%' THEN
            RETURN TRUE;
        END IF;
    END LOOP;
    RETURN FALSE;
END;
```

```
        END LOOP;  
        RETURN FALSE;  
END;
```

8 Scenariusze użycia

Projekt zawiera kompleksowe scenariusze testowe demonstrujące funkcjonalności systemu.

8.1 Scenariusz 1: Administrator rozszerza ofertę szkoły

Administrator dodaje nowy przedmiot (Flet), salę z wyposażeniem (VARRAY) oraz nauczyciela:

- `pkg_slovniki.dodaj_przedmiot('Flet', 'indywidualny')`
- `pkg_slovniki.dodaj_sale('105', 'indywidualna', 4, t_wyposazenie('Flet poprzeczny', 'Pulpit', 'Metronom'))`
- `pkg_osoby.dodaj_nauczyciela('Tomasz', 'Flecista', 6)` – REF do przedmiotu ID=6

8.2 Scenariusz 2: Sekretariat tworzy grupę i zapisuje uczniów

Sekretariat tworzy nową klasę 4A i zapisuje uczniów z referencją do grupy:

- `pkg_slovniki.dodaj_grupe('4A', 4)`
- `pkg_osoby.dodaj_ucznia('Jakub', 'Melodyjny', DATE '2014-07-20', 'Flet', 4)` – REF do grupy ID=4
- `pkg_osoby.lista_uczniow_grupy(4)` – używa kursora jawnego

8.3 Scenariusz 3: Planowanie lekcji grupowych

Sekretariat planuje zajęcia grupowe z walidacją typu sali i pojemności:

- `pkg_lekcje.dodaj_lekcje_grupowa(4, 4, 3, 4, DATE '2025-06-09', 17)`
- System waliduje: typ sali = 'grupowa', pojemność >= liczba uczniów, brak konfliktów

8.4 Scenariusz 4: Planowanie lekcji indywidualnych z konfliktem

Demonstracja walidacji konfliktów i heurystyki sugestii:

- **Sukces:** `pkg_lekcje.dodaj_lekcje_indywidualna(6, 6, 5, 10, DATE '2025-06-09', 14)`
- **Konflikt:** Próba dodania lekcji w zajęty terminie → błąd -20020 z sugestią alternatywnego terminu
- System przeszukuje VARRAY wyposażenia sal szukając pasującego instrumentu

8.5 Scenariusz 5: Nauczyciel wystawia oceny

Nauczyciel wystawia oceny z walidacją uprawnień:

- `pkg_oceny.wystaw_ocene(10, 6, 6, 5)` – ocena cząstkowa
- `pkg_oceny.wystaw_ocene_semestralna(10, 6, 6, 5)` – ocena semestralna
- **Błąd -20033:** Nauczyciel próbuje wystawić ocenę z przedmiotu którego nie uczy

8.6 Scenariusz 6: Uczeń sprawdza oceny i plan

Uczeń (lub rodzic) przegląda swoje dane:

- `pkg_oceny.oceny_ucznia(10)` – lista ocen z opisem słownym (metoda `opis_oceny()`)

- `SELECT pkg_oceny.srednia_ucznia(10, 6) FROM DUAL` – średnia z przedmiotu
- `pkg_lekcje.plan_ucznia(10)` – plan lekcji (UNION indywidualnych i grupowych)

8.7 Scenariusz 7: Raporty dla dyrekcji

Dyrekcja generuje raporty podsumowujące:

- `pkg_raporty.raport_grup()` – liczba uczniów w każdej klasie
- `pkg_raporty.statystyki()` – ogólne statystyki szkoły
- `pkg_lekcje.plan_nauczyciela(6)` – plan nauczyciela (kontrola obciążenia)

8.8 Scenariusz 8: Walidacje i przypadki brzegowe

Demonstracja walidacji i obsługi błędów:

- **Logika przypisania:** Lekcja bez ucznia i bez grupy → błąd -20001
- **Trigger ocen:** Ocena = 7 → błąd -20002
- **Typ sali:** Lekcja grupowa w sali indywidualnej → błąd -20031
- **Instrument:** Uczeń z Fletem na lekcji Fortepianu → błąd -20032
- **Kompetencje:** Nauczyciel Fletu prowadzi lekcję Fortepianu → błąd -20030

8.9 Scenariusz 9: Demonstracja VARRAY

Pokazanie działania kolekcji VARRAY wyposażenia sal:

- `pkg_slovniki.lista_sal()` – lista sal z wyposażeniem
- Metoda `lista_wyposazenia()` iteruje po VARRAY i zwraca tekst
- Funkcja `sala_ma_instrument()` przeszukuje VARRAY przy planowaniu

8.10 Scenariusz 10: Demonstracja REF/DEREF

Pokazanie działania referencji obiektowych:

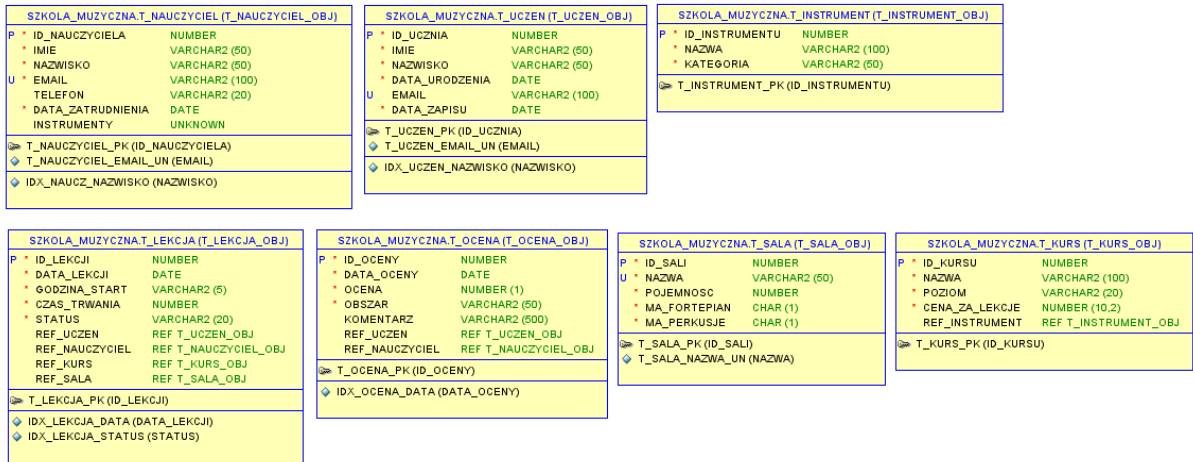
- Nauczyciele z przedmiotami (DEREF na REF do przedmiotu)
- Uczniowie z grupami (DEREF na REF do grupy)
- Lekcje z pełnymi danymi (wielokrotny Deref)

8.11 Scenariusz 11: Demonstracja metod obiektowych

Pokazanie działania metod zdefiniowanych w typach:

- `t_uczen.pelne_nazwisko()`, `t_uczen.wiek()`
- `t_nauczyciel.pelne_nazwisko()`, `t_nauczyciel.staz_lat()`
- `t_przedmiot.czy_grupowy()`, `t_sala.czy_grupowa()`
- `t_lekcja.czy_indywidualna()`, `t_lekcja.godzina_koniec()`
- `t_ocena.opis_oceny()`

9 Diagram relacji obiektów



Rysunek 1: Diagram relacji obiektów w bazie danych

10 Struktura plików projektu

Plik	Zawartość
01_typy.sql	Definicje 8 typów obiektowych z 10 metodami, VARRAY
02_tabele.sql	7 tabel obiektowych, 7 sekwencji
03_pakiety.sql	5 pakietów PL/SQL z 25 podprogramami publicznymi
04_triggery.sql	2 wyzwalacze walidacyjne (typ lekcji, zakres ocen)
05_dane.sql	Dane testowe (5 przedm., 3 grupy, 5 naucz., 4 sale, 9 uczn.)
06_testy.sql	Scenariusze testowe 1-11 (dokumentacja API + testy)
07_testy_rozszerzone.sql	Scenariusze testowe 12-23 (przypadki brzegowe)

Tabela 7: Pliki projektu

Kolejność uruchamiania:

1. 01_typy.sql – typy obiektowe z metodami
2. 02_tabele.sql – tabele obiektowe i sekwencje
3. 03_pakiety.sql – pakiety PL/SQL
4. 04_triggery.sql – wyzwalacze
5. 05_dane.sql – dane testowe
6. 06_testy.sql – scenariusze testowe (opcjonalne)
7. 07_testy_rozszerzone.sql – rozszerzone testy (opcjonalne)

11 Podsumowanie wymagań

Wymaganie	Status	Realizacja
Typy obiektowe z metodami	✓	8 typów, 10 metod (wiek, staz_lat, pelne_nazwisko, czy_grupowy, lista_wyposazenia, godzina_koniec, czy_indywidualna, opis_oceny)
Tabele obiektowe	✓	7 tabel obiektowych
REF i Deref	✓	9 referencji między obiektami
VARARRAY	✓	t_wyposazenie w tabeli sale + przeszukiwanie w sala_ma_instrument()
Pakiety PL/SQL	✓	5 pakietów, 25 podprogramów publicznych
Kursory	✓	Jawny (OPEN/FETCH/CLOSE) w lista_uczniow_grupy(), niejawny (FOR) w pozostałych
Obsługa błędów	✓	RAISE_APPLICATION_ERROR z 14 kodami błędów
Wyzwalacze	✓	2 triggerzy: relacja uczestników, zakres ocen
Walidacja konfliktów	✓	Funkcja prywatna sprawdz_kolizje()
Heurystyka	✓	Funkcja znajdz_alternatywe() (First Fit)

Tabela 8: Spełnienie wymagań projektu