**Sprawozdanie projektowe – zarządzanie cmentarzem**

1. **Wstęp**

Celem projektu było stworzenie obiektowej bazy danych, której zadaniem było przechowywanie informacji o pochowanych na cmentarzu zmarłych.

Cmentarz składa się ze stref na których pochowani należą do danej grupy wyznaniowej lub zawodowej (na przykład, ewangelicka, protestancka,   
wojskowa etc.).   
Każda strefa dzieliła się natomiast na sektory stanowiące mniejsze jednostki podziału terenu. W obrębie sektorów umieszczone zostały groby (pomniki), a każdy grób miał przypisany sobie sektor w którym był zlokalizowany.

Dodatkowo baza miała za zadanie przechowywać informacje o pogrzebach, zarówno tych co się odbyły jak i tych które dopiero zaplanowano.   
Do pogrzebów przypisane były również usługi z tym pogrzebem powiązane.

1. **Założenia projektowe**

W naszym projekcie przyjęliśmy następujące ograniczenia nałożone na bazę danych:

* Limit pogrzebów dla danego dnia – w ciągu jednego dnia mogą się odbyć maksymalnie **3** pogrzeby. W razie potrzeby istnieje jednak możliwość zmiany tej wartości z poziomu uprawnień administratora bazą cmentarza (użytkownik cemetery)
* Z każdym zmarłym powiązany jest tylko jeden pogrzeb.
* Zaplanowane pogrzeby dodawane za pomocą procedury add\_funeral nie mogą odbywać się w przeszłości. Wyjątkiem jest inicjalizacja bazy danych, podczas której dodawane są pogrzeby związane z leżącymi na cmentarzu zmarłymi. Jest ona jednak jedynie dostępna z perspektywy administratora bazy (użytkownik cemetery).
* Data urodzin pogrzebanego nie może być późniejsza niż data śmierci pogrzebanego
* Do każdego grobu (pomnika) przypisana jest dokładnie jedna opłata. Jest to najświeższa opłata powiązana z tym pomnikiem i ma ona związek z datą wygaśnięcia płatności za grób. Pozostałe opłaty związane z tym grobem archiwizowane są w tabeli płatności.
* Usługodawcy pogrzebowi nie mogą mieć więcej aktywnych zleceń niż 5. Wartość ta również jest definiowana przez zmienną i może być ustawiana przez administratora bazy danych.
* Usługodawcy pogrzebowi mogą przyjąć tylko jedno zlecenie danego dnia.
* Usługodawcy pogrzebowi nie mogą być 2 razy przypisani do tego samego pogrzebu.
* Usługodawca pogrzebowy może zostać przypisany do pogrzebu tylko wtedy gdy pogrzeb o przekazanym ID istnieje w tabeli pogrzebów.
* Aby dodać pochowanego do bazy danych, konieczne jest podanie takich ID pogrzebu i grobu, które znajdują się już w bazie danych (nie możemy pochować zmarłego w nieistniejącym grobie lub też przypisać mu nieistniejący pogrzeb).
* Aby dodać wpis grobu do bazy danych, konieczne jest podanie ID istniejącej, powiązanej z nim płatności. Sytuacja ma się analogicznie gdy aktualizujemy płatność za pomnik. Najpierw tworzona jest płatność, a później przypisywana jest ona do pomnika.
* Aby stworzyć nowy wpis o grobie, konieczne jest podanie identyfikatora referencji do istniejącego w bazie danych sektora.
* W grobie dowolnego typu może znajdować się dowolna liczba pochowanych
* Baza zezwala na tworzenie pogrzebów grupowych a zatem z jednym pogrzebem możemy powiązać wielu zmarłych.
* Nie ma limitu usługodawców pogrzebowych, których możemy przypisać do jednego pogrzebu.
* Nie jest sprawdzana integralność dat pomiędzy datą płatności, datą wygaśnięcia grobu i datą pogrzebu. Nie mniej jednak, zadbano o taką spójność w przypadku danych przykładowych wprowadzonych do bazy.

1. **Użytkownicy**

W projekcie zaimplementowano 3 typy użytkowników:

* 1. **Administrator bazy danych (cemetery)**

Użytkownik mający prawo do tworzenia obiektów, tabel, procedur, funkcji, etc. Jest on również właścicielem schematu cmentarza, a zatem ma dostęp do wszystkich elementów stworzonej bazy danych. Ma możliwość również jej rozbudowywania.

* 1. **Zarządca bazy danych (cemetery\_mgr)**Nie może nic tworzyć na schemacie bazy danych, ale ma dostęp do większości procedur i funkcji stworzonych przez administratora. Za ich pośrednictwem może umieszczać w bazie danych rekordy, usuwać je, modyfikować w pewnym stopniu, a także wyświetlać jej strukturę.
  2. **Gość (cemetery\_guest)**Rola odwiedzającego o stosunkowo wąskich uprawnieniach. Nie może nic tworzyć czy modyfikować a jedynie wyświetlać wpisy dotyczące pogrzebów, pomników, płatności etc. za pomocą przygotowanych procedur.

1. **Omówienie typów obiektowych**
   1. **sector\_t**

Typ opisujący sektor. Jego głównym celem jest odwzorowanie fizycznego podziału cmentarza w bazie a zatem składa się jedynie z identyfikatora sektora, oraz identyfikatora strefy, w której się znajduje

* 1. **payment\_t**

Typ opisujący płatność za grób. Przechowuje unikalny identyfikator płatności, datę transakcji oraz wielkość zapłaconej kwoty. Pozwala on na związanie poszczególnych opłat z grobami oraz na archiwizowanie transakcji w tym segmencie.

* 1. **grave\_t**  
     Typ opisujący grób. Posiada on referencję do sektora w którym się znajduje oraz referencję ostatniej płatności za miejsce na cmentarzu. Ponadto, przechowuje on również informacje o typie pomnika (ramka, kamienny, mauzoleum, urna), dacie wygaśnięcia płatności za miejsce oraz unikalny identyfikator.
  2. **funeral\_t**  
     Typ przechowujący dane nt. pogrzebu. Zawiera datę pogrzebu, nazwisko księdza z nim powiązanego oraz unikalny identyfikator pogrzebu.
  3. **service\_workers\_t**  
     Przechowuje informacje na temat usługodawców pogrzebowych (Imię, nazwisko, typ usługi, nr telefonu, koszt usługi, klasa usługodawcy, ilość oczekujących na realizację zamówień). Klasę usługodawcy określamy wartościami 1-3. Im mniejsza wartość, tym lepsza jest jakość zapewnianych usług.
  4. **service\_workers\_referencesNT**

Typ niezbędny do stworzenia tabeli zagnieżdżonej. Grupuje on referencje do obiektów typu service\_workers\_t

* 1. **buried\_t**Typ przeznaczony do przechowywania zmarłych. Zawiera on informacje o dacie urodzin i zgonu, imię, nazwisko, unikalny identyfikator zmarłego, a także referencje do grobu i do pogrzebu związanego z pochowanym.

Zalecane jest tworzenie typów zgodnie z kolejnością w jakiej zostały powyżej wypisane przez wzgląd na referencje zawarte w obiektach.

1. **Omówienie struktury tabel**

Relacje między tabelami prezentuje poniższy diagram relacji obiektów.

Obraz zawierający tekst, zrzut ekranu, diagram, linia

Opis wygenerowany automatycznie

Każda z tabel, poza tabelą FUNERALS\_AND\_SERVICES, jest tabelą obiektów odpowiedniego typu. Do stworzenia niektórych obiektów, potrzebne są jednak referencje do innych. Na przykład obiekt grave\_t posiada referencję do sektora i płatności której dotyczy a zatem tabela tychże obiektów będzie czerpała referencje z tabel PAYMENTS i SECTORS. Obiekty grave\_t będą z kolei posiadały referencję do obiektów grave\_t oraz funeral\_t,   
a zatem będą je zaciągały z tabel FUNERALS i GRAVES.

Tabela FUNERALS\_AND\_SERVICES różni się od pozostałych.   
Jej rekordy składają się z referencji typu funeral\_t oraz zagnieżdżonej tabeli typu service\_workers\_referencesNT, będącej tabelą referencji do obiektów typu service\_worker\_t. Taka struktura sprawia, że każdy pogrzeb ma własną listę referencji do usługodawców pogrzebowych z nim powiązanych. Tworzymy zatem relację jeden do wielu.

Zależność pomiędzy tabelą SECTORS i RELIGIOUS\_AREA została zaimplementowana w sposób relacyjny za pomocą wyzwalacza.

1. **Omówienie pakietów**

Każdemu typowi obiektowemu (za wyjątkiem typu tabeli zagnieżdżonej) przypisano pakiet zawierający procedury i funkcje związane z tym obiektem. Utworzono również 3 dodatkowe pakiety uzupełniające role zarządcy bazy i gościa.

* 1. **RELIGIOUS\_AREA\_UTILS** – pakiet związany ze strefami cmentarza
     1. **FUNCTION count\_graves(arg\_areaID IN NUMBER) RETURN NUMBER** - funkcja licząca groby w obrębie strefy arg\_areaID
     2. **FUNCTION count\_sectors(arg\_areaID IN NUMBER) RETURN NUMBER** - funkcja licząca sektory w obrębie strefy arg\_areaID
     3. **PROCEDURE display\_graves(arg\_areaID IN NUMBER)** - wyświetla informacje dotyczące wszystkich grobów znadujących się w obrębie strefy arg\_areaID
     4. **PROCEDURE display\_sectors(arg\_areaID IN NUMBER)** - wyświetla liczbę grobów w sektorach strefy arg\_areaID
  2. **SECTOR\_UTILS** – pakiet związany z sektorami cmentarza
     1. **FUNCTION count\_graves(arg\_sectorID IN NUMBER) RETURN NUMBER** – funkcja do liczenia grobów w obrębie sektora arg\_sectorsID
     2. **PROCEDURE change\_area(arg\_sectorID IN NUMBER, arg\_areaID NUMBER)** – pozwala zmienić strefę do której przypisany jest sektor arg\_sectorID.
     3. **PROCEDURE display\_graves(arg\_sectorID IN NUMBER)** – wyświetla informacje dotyczące wszystkich grobów w obrębie sektora arg\_sectorID
  3. **PAYMENT\_UTILS**
     1. **PROCEDURE add\_payment(arg\_paymentID NUMBER, value NUMBER, payment\_date DATE)** - dodanie nowej płatności do tabeli PAYMENTS.
     2. **PROCEDURE remove\_payment(arg\_paymentID NUMBER)** – wymazanie z tabeli PAYMENTS płatności o wskazanym w argumencie ID
     3. **PROCEDURE display\_payments()** – wyświetla wszystkie płatności z tabeli PAYMENTS
  4. **GRAVE\_UTILS / GRAVE\_FOR\_MANAGER** Zmienne:  
      - debug\_for\_archival\_data – zmienna filtrująca: 1 – można dodawać nieopłacone groby, 0 – nie ma takiej możliwości  
     1. **PROCEDURE update\_expiration\_date(arg\_graveID in NUMBER,new\_date in DATE)** – aktualizuje datę wygaśnięcia opłaty za grób arg\_graveID
     2. **PROCEDURE update\_grave\_type(arg\_graveID in NUMBER,new\_type in VARCHAR) –** pozwala dostosować rodzaj grobu. Zmienna new\_type jest ustandaryzowana i zezwala jedynie na wprowadzenie typów: stone,frame,mauzoleum i urn. W przeciwnym razie procedura zwróci błąd.
     3. **PROCEDURE assign\_buried(arg\_graveID in NUMBER, buriedID in NUMBER) –** przypisuje zmarłego o ID arg\_buriedID do grobu o ID arg\_graveID. Jeżeli nie istnieje grób lub zmarły o wskazanym ID, to zwracany jest błąd.
     4. **PROCEDURE show\_buried(arg\_graveID IN NUMBER)** – wyświetla wszystkich zmarłych pochowanych w grobie arg\_graveID.
     5. **PROCEDURE update\_payment(arg\_paymentID IN NUMBER, arg\_graveID IN NUMBER)** – procedura do aktualizowania opłaty za grób arg\_graveID. Przypisywana jest referencja do nowej opłaty, a stara opłata pozostaje w tabeli PAYMENTS jako rekord archiwalny.   
        Zaleca się stosowanie wraz z tą procedurą procedury update\_expiration\_date.
     6. **PROCEDURE add\_grave(arg\_graveID IN NUMBER, expiration\_date IN DATE, grave\_type IN VARCHAR, arg\_paymentID IN NUMBER, arg\_sectorID IN NUMBER)** – procedura dodania grobu do tabeli GRAVES wraz ze sprawdzeniem ograniczeń. Istnienie obiektów płatności i sektora wskazanych przez przekazane w funkcji ID sprawdza wyzwalacz.
     7. **PROCEDURE remove\_grave(arg\_graveID IN NUMBER) -** procedura usunięcia grobu z tabeli GRAVES wraz ze sprawdzeniem ograniczeń.
     8. **PROCEDURE remove\_grave\_and\_buried(arg\_graveID IN NUMBER)** - procedura usunięcia grobu z tabeli GRAVES wraz ze sprawdzeniem ograniczeń, ale wraz z usunięciem z tabeli BURIED wszystkich pogrzebanych w nim zmarłych.
     9. **\*FUNCTION get\_debug\_for\_archival\_data RETURN NUMBER –** pobranie wartości zmiennej filtrującej wprowadzane dane
     10. **\*PROCEDURE set\_debug\_for\_archival\_data(new\_value NUMBER) –** ustawienie zmiennej odpowiadającej za wprowadzanie do tabeli grobów których expiration\_date już wygasło.   
         1 – filtrowanie grobów nieopłaconych wyłączone  
          0 - filtrowanie grobów nieopłaconych włączone  
           
         \* - procedury i funkcje dostępne dla użytkownika CEMETERY, ale niewystępujące w pakiecie GRAVES\_FOR\_MANAGER.
  5. **FUNERAL\_UTILS / FUNERAL\_FOR\_MANAGER**Zmienne:  
     - max\_funerals\_per\_day – limit pogrzebów jakie mogą mieć miejsce w ciągu jednego dnia (domyślnie 3)  
     - max\_orders\_per\_worker – limit aktywnych zamówień jakie może mieć dany usługodawca pogrzebowy (domyślnie)  
     - debug\_for\_archival\_data – zmienna filtrująca: 1 – można wprowadzać pogrzeby które już się odbyły, 0 – nie ma takiej możliwości  
     1. **PROCEDURE add\_funeral(arg\_funeralID NUMBER, arg\_funeral\_date DATE, arg\_priest\_name VARCHAR)** – dodaje pogrzeb do tabeli FUNERALS oraz FUNERALS\_AND\_SERVICES wraz ze sprawdzeniem ograniczeń.
     2. **FUNCTION calculate\_funeral\_cost(arg\_funeralID NUMBER) RETURN NUMBER –** oblicza koszt pogrzebu na podstawie sumy ceny usług przypisanych do niego pracowników pogrzebowych.
     3. **FUNCTION remove\_funerals\_by\_date(cutoff\_date DATE) RETURN NUMBER –** usuwa z tabeli FUNERALS wszystkie pogrzeby, które miały miejsce wcześniej niż data podana w parametrze cuttof\_date.
     4. **PROCEDURE assign\_service\_worker(arg\_funeralID NUMBER, arg\_workerID NUMBER) –** przypisuje do pogrzebu usługodawcę pogrzebowego. Funkcja sprawdza również ograniczenia bazy w tym kontekście (czy istnieje pogrzeb i usługodawca, czy usługodawca może przyjąć kolejne zlecenie, czy nie jest już przypisany do tego pogrzebu).
     5. **PROCEDURE remove\_service\_worker(arg\_funeralID NUMBER, arg\_workerID NUMBER) –** usuwa usługodawcę z pogrzebu (pod warunkiem że usługodawca i pogrzeb istnieją). Usługodawca pozostaje w tabeli SERVICE\_WORKERS.
     6. **PROCEDURE display\_funerals() –** wyświetla informacje dotyczące kolejnych pogrzebów z tabeli FUNERALS
     7. **\*FUNCTION get\_max\_funerals\_per\_day RETURN NUMBER -**
     8. **\*FUNCTION get\_max\_orders\_per\_worker RETURN NUMBER**
     9. **\*FUNCTION get\_debug\_for\_archival\_data RETURN NUMBER**
     10. **\*PROCEDURE set\_max\_funerals\_per\_day(new\_value NUMBER)**
     11. **\*PROCEDURE set\_max\_orders\_per\_worker(new\_value NUMBER)**
     12. **\*PROCEDURE set\_debug\_for\_archival\_data(new\_value NUMBER)**
     13. **PROCEDURE remove\_funeral\_by\_id(funeral\_to\_remove NUMBER) –** usuwa z tabeli FUNERALS pogrzeb o konkretnym ID.  
           
         \* funkcje g-l to gettery i settery do zmiennych filtrujących i ograniczających. Ma do nich dostęp jedynie użytkownik CEMETERY. Nie znajdują się one w pakiecie FUNERAL\_FOR\_MANAGER. Informacje nt. zmiennych zawarto powyżej.
  6. **SERVICE\_WORKER\_UTILS**
     1. **PROCEDURE add\_service\_worker(arg\_workerID NUMBER, name VARCHAR, surname VARCHAR, profession VARCHAR,phone VARCHAR, reputation NUMBER, service\_cost NUMBER) –** dodaje usługodawcę pogrzebowego do tabeli SERVICE\_WORKERS.
     2. **PROCEDURE remove\_service\_worker(arg\_workerID NUMBER)** – usuwa usługodawcę pogrzebowego za tabeli SERVICE\_WORKERS.
     3. **FUNCTION count\_orders\_in\_future(arg\_workerID NUMBER) RETURN NUMBER –** funkcja licząca aktywne zamówienia dla pracownika o wskazanym ID (nie są uwzględniane pogrzeby które już się odbyły).
     4. **PROCEDURE update\_orders\_count\_by\_ID(arg\_workerID NUMBER)** – funkcja aktualizująca aktywne zamówienia dla danego pracownika. Aktualizacja czyli ponowne sumowanie zamówień tylko z tych pogrzebów, które się jeszcze nie odbyły.
     5. **PROCEDURE update\_orders\_count\_for\_all\_workers –** aktualizacja aktywnych zamówień dla wszystkich pracowników z tabeli SERVICE\_WORKERS. Zalecane jest regularne stosowanie funkcji (np. kilka razy na dzień) w celu zachowania poprawności danych.
     6. **PROCEDURE display\_service\_workers()** – wyświetla informacje nt. wszystkich pracowników znajdujących się w tabeli SERVICE\_WORKERS.
  7. **BURIED\_UTILS**
     1. **function calculate\_age(buried in buried\_t) return number** – funkcja obliczająca wiek zmarłego.
     2. **procedure create\_buried(arg\_personID in number, name in VARCHAR, surname in VARCHAR,born in DATE,passed in DATE, arg\_graveID in NUMBER, arg\_funeralID in NUMBER)** – tworzy wpis o zmarłym sprawdzając jednocześnie ograniczenia bazy danych. Poprawność referencji definiuje odpowiedni trigger.
     3. **procedure remove\_buried(buried\_id in number) –** funkcja kasująca wpis o zmarłym z tabeli BURIED jeśli znajdował się w niej wpis o wskazanym ID.
     4. **PROCEDURE display\_buried(arg\_funeralID IN NUMBER) –** wyświetla informacje dotyczące zmarłych związanych ze wskazanym pogrzebem. Jeżeli arg\_funeralID zostanie ustawione na 0, to wyświetleni zostaną wszyscy pogrzebani.
  8. **GUEST\_PACKAGE**
     1. **PROCEDURE display\_graves\_in\_area(arg\_areaID IN NUMBER)** – wyświetla groby we wskazanej strefie.
     2. **PROCEDURE display\_graves\_in\_sector(arg\_sectorID IN NUMBER)** – wyświetla groby we wskazanym sektorze.
     3. **PROCEDURE display\_payments()** – wyświetla WSZYSTKIE płatności za groby.
     4. **PROCEDURE show\_buried\_in\_grave(arg\_graveID IN NUMBER)** – wyświetla zmarłych pochowanych we wskazanym grobie.
     5. **PROCEDURE display\_funerals()** – wyświetla WSZYSTKIE pogrzeby.
     6. **PROCEDURE display\_service\_workers()** – wyświetla WSZYSTKICH usługodawców pogrzebowych.
     7. **PROCEDURE display\_buried\_in\_funeral(arg\_funeralID IN NUMBER)** – wyświetla pogrzebanych powiązanych ze wskazanym pogrzebem. Jeżeli arg\_funeralID zostanie ustawione na 0, to wyświetleni zostaną wszyscy pogrzebani.

1. **Omówienie wyzwalaczy**
   1. TRIGGER **ADD\_BURIED\_INTEGRITY\_TRIGGER** BEFORE INSERT OR UPDATE ON BURIED FOR EACH ROW – wyzwalacz sprawdzający przy przypisywaniu nowego zmarłego czy istnieją wskazane za pomocą ID referencje do pogrzebu i grobu.
   2. TRIGGER **ADD\_GRAVE\_INTEGRITY\_TRIGGER** BEFORE INSERT OR UPDATE ON GRAVES FOR EACH ROW – wyzwalacz sprawdzający przy tworzeniu nowego grobu czy istnieją wskazane za pomocą ID referencje do płatności i sektora.
   3. TRIGGER **KEEP\_GRAVE\_COUNT\_IN\_SECTOR** BEFORE DELETE OR INSERT OR UPDATE OF sector\_ref ON GRAVES FOR EACH ROW – trigger odpowiadający za liczenie grobów w obrębie sektorów. Wartości count są zmieniane ilekroć nastąpi operacja dodania, usunięcia bądź też modyfikacji grobu.
   4. TRIGGER **SECTORS\_TO\_AREAID\_FOREIGN\_KEY\_TRIGGER** BEFORE INSERT OR UPDATE OF AREAID, GRAVE\_COUNT ON SECTORS FOR EACH ROW – sektor będący kluczem obcym na zmiennej areaID w obiektach typu sektor\_t związanym z polem areaID w tabeli RELIGIOUS\_AREAS. Sprawdza czy wpisywane do zmiennej areaID w obiektach sektor\_t wartości mają odzwierciedlenie w tabeli RELIGIOUS\_AREAS.
2. **Zastosowane mechanizmy**

W bazie danych zastosowano:

1. Pakiety – do grupowania funkcji i procedur
2. Wyzwalacze – do sprawdzania poprawności danych
3. Kursory – głównie do wyświetlania danych
4. Ref kursor – zastosowany w funkcjach display\_buried\_in\_funeral oraz display\_buried pozwala na zmianę trybu wyświetlania w zależności od wartości parametru wejściowego.
5. Tabele zagnieżdżone – w tabeli FUNERALS\_AND\_SERVICES. Przechowują referencje do usługodawców związanych z konkretnymi pogrzebami (relacja jeden do wielu).
6. Typy obiektowe
7. Tabele zbudowane z typów obiektowych
8. Referencje do obiektów (zawarte w typach obiektowych)
9. Dereferencje służące do wyłuskiwania wartości z referencyjnych pól obiektów.