Założenia projektu zaliczeniowego

Cel:
Kompletne opracowanie projektu zaliczeniowego, na który składa się implementacja w środowisku Oracle modelu relacyjnej bazy danych, oprogramowanie realizujące proces automatycznego generowania danych wejściowych oraz prezentujące wyniki w postaci analitycznej.

Harmonogram projektu:

1. Opracowanie modelu bazy danych, w oparciu o który realizowane będą dalsze części projektu.
2. Implementacja modelu na udostępnionym serwerze Oracle oraz opracowanie programowego symulatora generującego dane demonstracyjne.
3. Opracowanie skryptów wdrożeniowych służących do implementacji i zarządzania projektem.
4. Opracowanie dokumentacji projektu.

Termin zakończenia projektu:

Termin zasadniczy - trzy dni przed ostatnimi zajęciami.

Szczegółowe założenia projektowe:

1. Opracować model bazy danych dowolnego fragmentu rzeczywistości zawierający elementy transakcyjne (sprzedaż, wypożyczenie, naprawa itp.). Dodatkowo strony transakcji podlegają wymiarowaniu poprzez tabele słownikowe (grupy produktowe, województwa, marki samochodów itp.). Do realizacji tego punktu należy wykorzystać narzędzie Oracle Data Modeler. Model powinien być zaimplementowany na serwerze Oracle 12c (city.wsisiz.edu.pl). Model logiczny powinien zawierać do 10 encji będących w związkach M:N (obowiązkowo) i 1:N, w którym występują atrybuty służące do liczenia (koszty, zyski, punkty itp.).

2. Implementacja powinna zawierać dwie metody generowania kluczy głównych tabel, tzn. sekwencje i wyzwalacze. Dodatkowo wyzwalacze powinny zawierać elementy zautomatyzowanych obliczeń zgodnie ze swoim przeznaczeniem, zapewniając spójność bazy danych.

3. Do wszystkich tabel należy wprowadzić sensowne dane.

4. Opracować w postaci minimum czterech perspektyw wyniki zbiorcze obrazujące efekty działania zaprojektowanego oprogramowania na podstawie wprowadzonych danych. Perspektywy powinny pokazywać typowe wskaźniki biznesowe, na przykład zyski z prowadzonej działalności, wielkość sprzedaży poszczególnych produktów w funkcji czasu (lata, kwartały czy miesiące). Perspektywy powinny być zbudowane w sensowny sposób w oparciu o co najmniej trzy tabele. Można wykorzystać również zmienne wiązania symulując działanie w aplikacji. Na podstawie wybranych dwóch perspektyw należy utworzyć raporty w środowisku JasperReports (typu zbiorczego zestawienia transakcji oraz analitycznego odpowiednio sparametryzowanych).

5. Opracować sparametryzowane trzy funkcje służące celom obliczeniowym (w tym jedną logiczną) i trzy procedury zapewniające wprowadzanie, aktualizację i kasowanie danych w bazie. Oprogramowanie to powinno zawierać walidację danych i generować stosowne komunikaty końcowe.

6. Opracować skrypty wdrożeniowe umożliwiające instalację i deinstalację projektu na dowolnym koncie. Skrypt instalujący powinien zawierać zdania SQL tworzące obiekty bazodanowe (tabele, sekwencje, perspektywy, wyzwalacze, procedury i funkcje) oraz zdania wprowadzające dane.

Dopuszcza się możliwość istnienia kilku skryptów, na przykład tworzenie obiektów i wprowadzanie danych do tabel. Skrypt deinstalujący powinien usunąć ze schematu cały projekt.

7. W postaci zdań select zademonstrować użycie perspektyw, zdań podrzędnych nieskorelowanych umieszczonych we frazach from, where i having i zawierających funkcje agregujące i limitowanie wierszy (rownum). Można zastosować zmienne wiązania.

8. Opracować dokumentację projektową zawierającą:

   8.1. Analizę biznesową projektowanej rzeczywistości,
   8.2. Model logiczny i relacyjny bazy danych,
   8.3. Oprogramowanie tworzące bazę danych,
   8.4. Skrypty wdrożeniowe instalujące i deinstalujące zrealizowany projekt,
   8.5. Instrukcję instalacji projektu i sprawdzenia jego poprawności,
   8.6. Wyniki działania perspektyw w postaci raportów pdf.

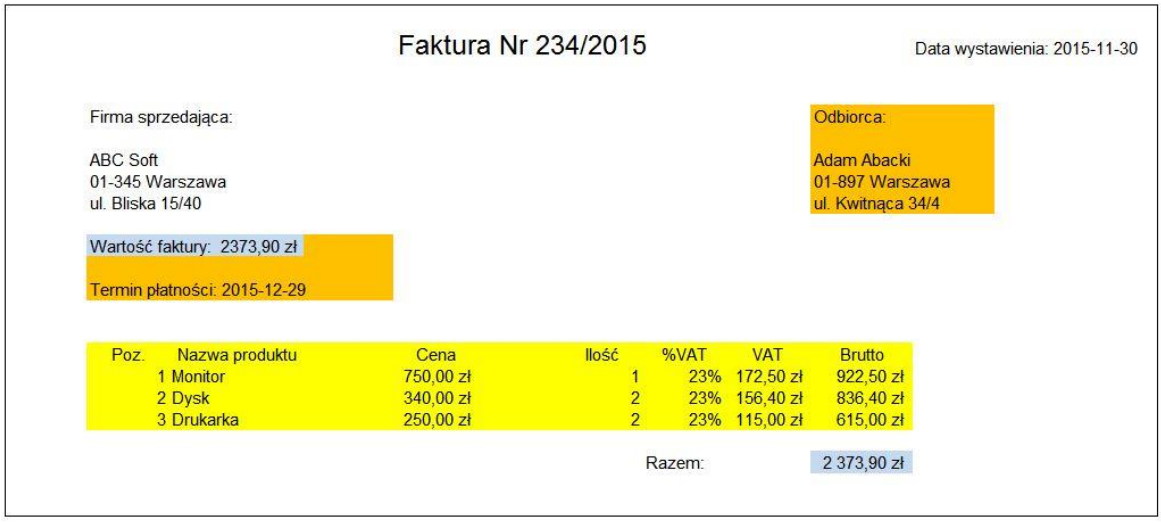
Dodatkowe założenia (nieobowiązkowe):

9. Opracować programowy generator danych przy pomocy języka PL/SQL. Generator powinien w sposób automatyczny lub manualny generować transakcje na podstawie wprowadzonych danych stałych (strony transakcji i dane słownikowe). Zbudowany powinien być ze sparametryzowanych procedur i funkcji PL/SQL (ewentualnie zawierających zmienne wiązania). Do realizacji tego punktu w sposób automatyczny można wykorzystać obiekty programowe *jobs*.

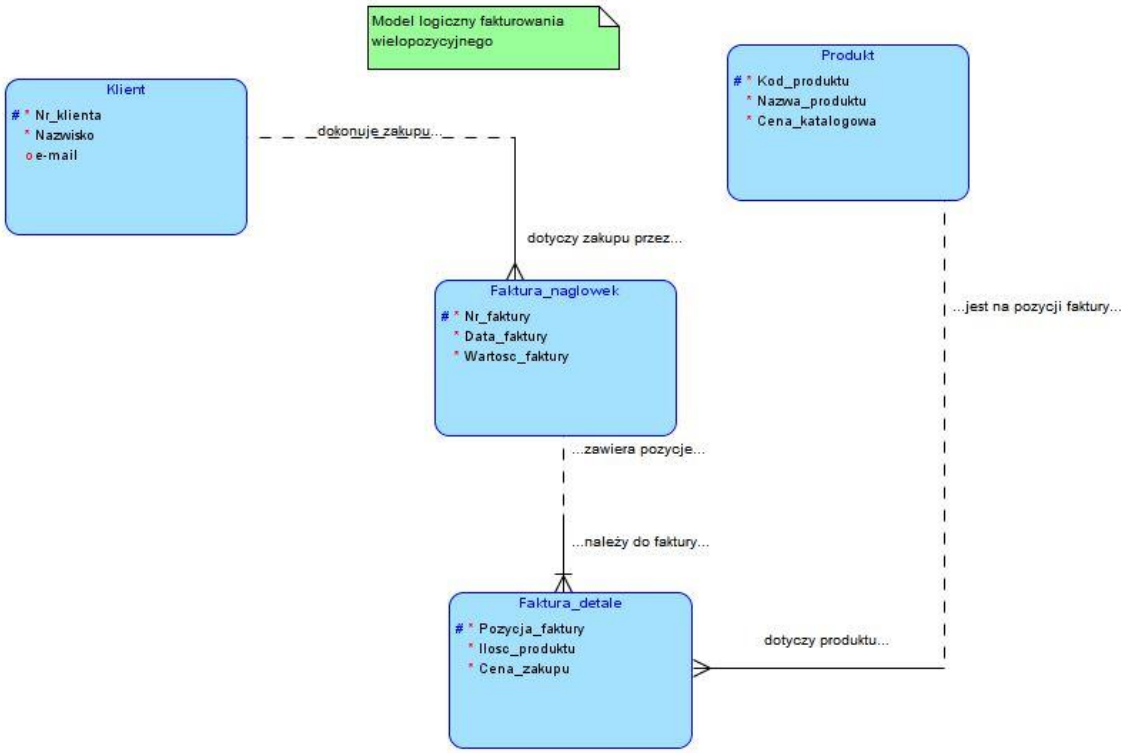
10. Zastępczo, zamiast tworzenia raportów przy pomocy Jasper Reports i programowego generatora danych, można wykonać fragment aplikacji w Oracle Application Express. Wymaga to wcześniejszego uzgodnienia w celu przygotowania środowiska programowego oraz omówienia założeń na tworzoną aplikację.

Materiały pomocnicze:

Typowa faktura zakupowa składa się z elementów przedstawionych na poniższym rysunku:

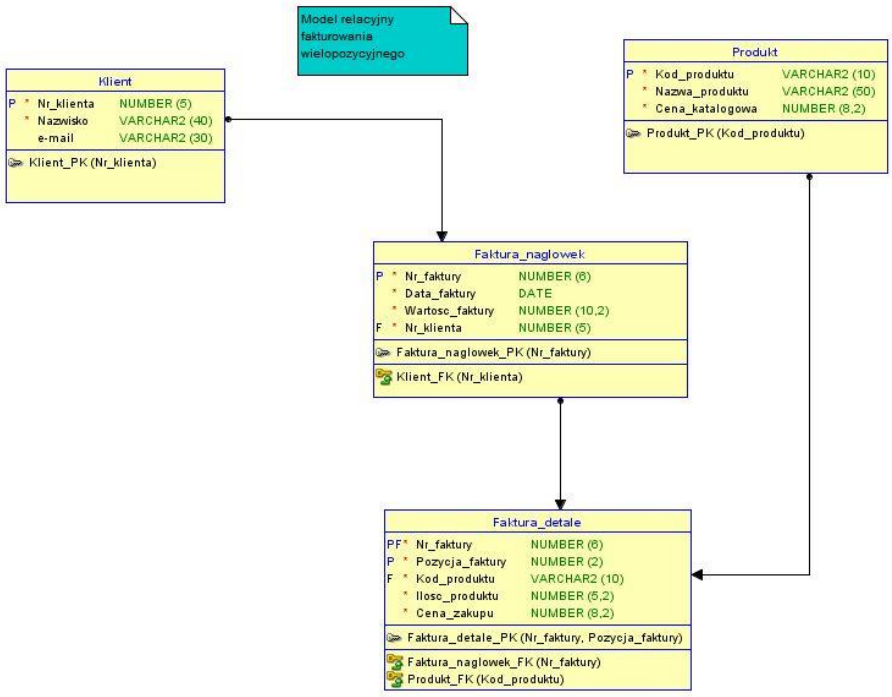


Typowy model fakturowania wielopozycyjnego dotyczący sprzedaży towarów zarejestrowanym klientom można przedstawić poniższym modelem logicznym:

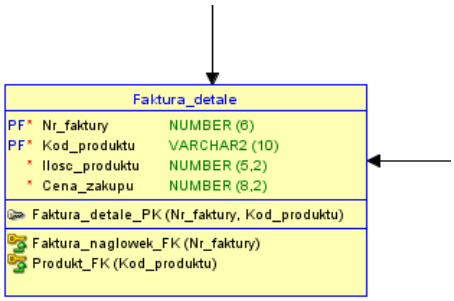


Model ten zawiera dwie encje będące stronami transakcji (Klient i Produkt) oraz dwie (pozostałe) przedstawiające model faktury wielopozycyjnej. Wszystkie związki zaznaczone na powyższym diagramie są typu 1:N. Klient może mieć wystawionych wiele faktur, produkt może występować na wielu pozycjach faktur, a jedna faktura może mieć wiele pozycji.

Odpowiadający powyższemu relacyjny model wyglądać będzie tak:



Drugim wariantem tego modelu może być brak pozycji faktury i wtedy kluczem głównym tabeli Faktura_detale będzie numer faktury i kod produktu.



Pierwsze rozwiązanie ma tę zaletę, że kolejność pozycji na fakturze jest określana przez operatora, na przykład według ważności pozycji (przy kupnie samochodu najpierw samochód, a potem płyn do spryskiwacza). Wadą jest możliwość zdublowania danej pozycji na fakturze. Drugie rozwiązanie eliminuje możliwość wystąpienia powtarzających się pozycji, ale kolejność na fakturze może być przypadkowa i zależna od kodu produktu.

Założenia alternatywne (dla studentów studiów zaocznych):

1. Założenia na system informatyczny (pod kątem modelu bazy danych):

System informatyczny do obsługi ośrodka szkolenia kierowców jest oparty na bazie danych, w której przechowywane są dane kursantów – m.in. dane personalne, wybrana kategoria prawa jazdy, preferencje dotyczące lokalizacji zajęć, listy dostarczonych dokumentów oraz dokonanych i zaległych opłat; dane instruktorów – m.in. dane personalne, preferowana lokalizacja zajęć, informacja na temat możliwości prowadzenia zajęć praktycznych i wykładów teoretycznych dla poszczególnych kategorii prawa jazdy oraz dane samochodów – m.in. dane techniczne, informacja o możliwości prowadzenia zajęć dla poszczególnych kategorii prawa jazdy, informacje o naprawach oraz planowanych przeglądach. Należy stworzyć możliwość monitowania o zbliżających się terminach obowiązkowych przeglądów oraz o konieczności odbioru naprawionych samochodów.
System powinien przechowywać również informacje o planowanych terminach kursów oraz udostępniać informacje o wolnych terminach szkoleń. Zapisy na kurs mogą odbywać się sposobem tradycyjnym lub mailowo. Jeśli kursant poda swój adres email, każda rejestracja będzie automatycznie potwierdzana drogą mailową.
W systemie mają być przechowywane również dodatkowe informacje na temat przeprowadzonych kursów, na przykład przechowywane będą sugestie uczestników kursów w postaci ankiet na temat ich przebiegu oraz możliwych usprawnień.
Po przeprowadzeniu każdego egzaminu wewnętrznego zapisywany będzie jego wynik oraz w przypadku wyniku negatywnego – powód niezaliczenia. Zakłada się, że ośrodek będzie otrzymywał z zewnętrznego źródła wyniki egzaminów państwowych swoich kursantów. Wyniki te będą wprowadzane i przechowywane w systemie. Na podstawie tych danych prowadzone będą statystyki najczęstszych problemów, na podstawie których będzie można dostosować optymalny plan kursów.

2. Przy pomocy Oracle Data Modeler opracować model logiczny i relacyjny projektowanego systemu.

3. W modelu opracować perspektywy dedykowane osobom funkcyjnym ośrodka, takim jak *rejestratorka kursantów, nadzorca samochodów* i *kierownik ośrodka, kursant*.

4. Wykonać dokumentację projektu, na którą składać się będą założenia na projektowany system, przyjęte rozwiązania w postaci modelu logicznego i relacyjnego wraz ze stosownym opisem oraz wygenerowany skrypt realizujący implementację projektu na jednym z możliwych serwerów bazodanowych.