Załącznik Nr 2 do SOPZ

Szczegółowy zakres minimalnych wymagań podsystemów dla dostarczenia i konfiguracji oprogramowania bazodanowego

1. Przedmiot zamówienia:
Dostarczenie licencji, instalacja i konfiguracja oprogramowania bazodanowego.

2. Minimalne wymagania:

System bazodanowy (SBD)

Lp.	Opis wymagania
1.	Dostępność oprogramowania na współczesne 64-bitowe platformy Unix (HP-UX dla procesorów Itanium, Solaris dla procesorów SPARC i Intel/AMD, IBM AIX dla procesorów POWER, Intel/AMD Linux, MS Windows). Identyczna funkcjonalność serwera bazy danych na ww. platformach
2.	Dostarczone licencje nie mogą ograniczać liczby użytkowników końcowych korzystających z oprogramowania ani liczby przetwarzanych lub przechowywanych dokumentów, plików, rekordów, żądań, etc. Licencje nie mogą być ograniczone czasowo.
3.	Proponowany zestaw licencji powinien być jednorodny. Wymagana jest dostawa oprogramowania certyfikowanego pod względem zgodności ze sobą. Wymaganie obejmuje: · Oprogramowanie bazy danych ze względu na zgodność z systemem operacyjnym oraz platformą sprzętową, · Systemy operacyjne używane do uruchamiania serwerów bazy danych ze względu na zgodność z platformą sprzętową.
4.	Dostępność narzędzi migracji baz danych pomiędzy platformami na poziomie fizycznym (kopiowanie / konwersja plików danych) oraz logicznym (narzędzia eksportu / importu).
5.	Oprogramowanie klienckie, za pomocą którego można łączyć się do bazy danych musi być dostępne na wielu platformach systemowo-sprzętowych (minimalny zakres platform taki jak dla oprogramowania serwera bazy danych).
6.	Wsparcie protokołu XA.
7.	Wsparcie standardu JDBC 3.0.
8.	Zgodność ze standardem ANSI/ISO SQL 2003 lub nowszym.
9.	Wbudowana obsługa wyrażeń regularnych zgodna ze standardem POSIX dostępna z poziomu języka SQL jak i procedur/funkcji składowanych w bazie danych.
10.	RDBMS musi zapewniać niezależność platformy systemowej dla oprogramowania klienckiego od platformy systemowej bazy danych.
11.	RDBMS musi zapewniać przetwarzanie transakcyjne wg reguł ACID z zachowaniem spójności i maksymalnego możliwego stopnia współbieżności. Mechanizm izolowania transakcji musi pozwalać na spójny odczyt modyfikowanego obszaru danych bez wprowadzania blokad, spójny odczyt nie może blokować możliwości wykonywania zmian.
12.	RDBMS musi posiadać możliwość zagnieżdżania transakcji – możliwość

	uruchomienia niezależnej transakcji wewnątrz transakcji nadrzędnej.
13.	Dostępność nieblokującego poziomu izolowania transakcji „tylko do odczytu” (Read Only) pozwalający na uzyskanie w wielu kolejnych następujących po sobie zapytaniach rezultatów odzwierciedlających stan danych z chwili rozpoczęcia ww. transakcji.
14.	Dostępność poziomu serializowanego poziomu izolowania transakcji (Serializable).
15.	Możliwość zmiany domyślnego trybu izolowania transakcji (Read Commited) na inny (Read Only, Serializable) za pomocą komend serwera bazy danych.
16.	Wsparcie dla wielu ustawień narodowych i wielu zestawów znaków (włącznie z Unicode) zarówno po stronie serwera bazy danych jak i oprogramowania klienckiego. Wsparcie dla polskich stron kodowych – ISO-8859-2, MS Windows Code Page 1250 oraz PC 852. Automatyczna konwersja znaków pomiędzy różnymi ustawieniami stron kodowych po stronie klienta i serwera bazy danych.
17.	Możliwość migracji bazy danych utrzymujących dane znakowe w 8-bitowej stronie kodowej do Unicode.
18.	Możliwość definiowania w przestrzeni danych (plików) dla danych użytkownika obszarów o innym niż domyślny rozmiarze bloku.
19.	Możliwość bez dodatkowych ograniczeń przechowywania wierszy, których rozmiar przekracza rozmiar bloku bazy danych.
20.	Możliwość budowania indeksów o strukturze B-drzewa. Baza danych powinna umożliwiać założenie indeksu jednej lub większej liczbie kolumn tabeli, przy czym ograniczenie liczby kolumn na których założony jest 1 indeks nie powinno być mniejsze niż 16.
21.	Możliwość budowania widoków zmaterializowanych odzwierciedlających stan danych zdefiniowanych przez zapytanie SQL. Widok zmaterializowany przechowuje rezultat zapytania, którego aktualizacja odbywa się w jednej z dostępnych strategii – na żądanie, okresowo bądź po każdym zatwierdzeniu transakcji modyfikującej tabele, na której oparty jest widok zmaterializowany.
22.	Możliwość szybkiego odświeżania danych w widoku zmaterializowanym na podstawie mechanizmu identyfikacji zmian w danych źródłowych.
23.	Brak formalnych ograniczeń na liczbę tabel i indeksów w bazie danych oraz na ich rozmiar (liczbę wierszy).
24.	Kosztowy model optymalizacji instrukcji SQL.
25.	Model statystyk optymalizatora kosztowego musi pozwalać na odwzorowanie nierównomierności rozkładu danych (składowanie informacji o rozkładzie wartości występujących w kolumnach za pomocą histogramu bądź porównywalnego funkcjonalnie modelu odwzorowania).
26.	Możliwość uwzględnienia korelacji wartości występujących w niezależnych kolumnach tabeli w modelu statystyk optymalizatora kosztowego.
27.	RDBMS powinien umożliwiać wskazywanie optymalizatorowi SQL preferowanych metod optymalizacji na poziomie konfiguracji parametrów pracy serwera bazy danych oraz dla wybranych zapytań. Powinna istnieć możliwość umieszczania wskazówek dla optymalizatora w wybranych instrukcjach SQL.
28.	Wsparcie dla procedur i funkcji składowanych w bazie danych. Język programowania powinien być językiem proceduralnym, blokowym (umożliwiającym deklarowanie zmiennych wewnątrz bloku), oraz wspierającym obsługę wyjątków. W przypadku, gdy wyjątek nie ma zadeklarowanej obsługi wewnątrz bloku, w razie jego wystąpienia wyjątek powinien być automatycznie propagowany do bloku nadrzędnego bądź wywołującej go jednostki programu.
29.	Procedury i funkcje składowane powinny mieć możliwość parametryzowania za

	pomocą parametrów prostych jak i parametrów o typach złożonych, definiowanych przez użytkownika. Funkcje powinny mieć możliwość zwracania rezultatów jako zbioru danych, możliwego do wykorzystania jako źródło danych w instrukcjach SQL (czyli występujących we frazie FROM). Ww. jednostki programowe powinny umożliwiać wywoływanie instrukcji SQL (zapytania, instrukcje DML, DDL), umożliwiać jednoczesne otwarcie wielu tzw. kursorów pobierających paczki danych (wiele wierszy za jednym pobraniem) oraz wspierać mechanizmy transakcyjne (np. zatwierdzanie bądź wycofanie transakcji wewnątrz procedury).
30.	Możliwość kompilacji procedur składowanych w bazie do postaci kodu binarnego (biblioteki dzielonej).
31.	Możliwość deklarowania wyzwalaczy (triggerów) na poziomie instrukcji DML (INSERT, UPDATE, DELETE) wykonywanej na tabeli, poziomie każdego wiersza modyfikowanego przez instrukcję DML oraz na poziomie zdarzeń bazy danych (np. próba wykonania instrukcji DML, start serwera, stop serwera, próba zalogowania użytkownika, wystąpienie specyficznego błędu w serwerze). Ponadto mechanizm wyzwalaczy powinien umożliwiać oprogramowanie obsługi instrukcji DML (INSERT, UPDATE, DELETE) wykonywanych na tzw. niemodyfikowalnych widokach (views).
32.	W przypadku, gdy w wyzwalaczu na poziomie instrukcji DML wystąpi błąd zgłoszony przez motor bazy danych bądź ustawiony wyjątek w kodzie wyzwalacza, wykonywana instrukcja DML musi być automatycznie wycofana przez serwer bazy danych, zaś stan transakcji po wycofaniu musi odzwierciedlać chwilę przed rozpoczęciem instrukcji w której wystąpił ww. błąd lub wyjątek.
33.	Możliwość wykonania równoczesnych operacji DML (Insert/Update/Delete) na tej samej tabeli .
34.	Powinna istnieć możliwość autoryzowania użytkowników bazy danych za pomocą rejestru użytkowników założonego w bazie danych bądź mechanizmu zewnętrznego w stosunku do bazy danych.
35.	Przywileje użytkowników bazy danych powinny być określane za pomocą przywilejów systemowych (np. prawo do podłączenia się do bazy danych - czyli utworzenia sesji, prawo do tworzenia tabel itd.) oraz przywilejów dostępu do obiektów aplikacyjnych (np. odczytu / modyfikacji tabeli, wykonania procedury). Baza danych powinna umożliwiać nadawanie ww. przywilejów za pośrednictwem mechanizmu grup użytkowników / ról bazodanowych. W danej chwili użytkownik może mieć aktywny dowolny podzbiór nadanych ról bazodanowych.
36.	Możliwość wykonywania i katalogowania kopii bezpieczeństwa bezpośrednio przez serwer bazy danych. Możliwość zautomatyzowanego usuwania zbędnych kopii bezpieczeństwa przy zachowaniu odpowiedniej liczby kopii nadmiarowych - stosownie do założonej polityki nadmiarowości backup'ów. Możliwość integracji z powszechnie stosowanymi systemami backupu (Legato, Veritas, Tivoli, itp.). Wykonywanie kopii bezpieczeństwa powinno być możliwe w trybie offline oraz w trybie online(hot backup).
37.	Odtwarzanie powinno umożliwiać odzyskanie stanu danych z chwili wystąpienia awarii bądź cofnąć stan bazy danych do punktu w czasie. W przypadku odtwarzania do stanu z chwili wystąpienia awarii odtwarzaniu może podlegać cała baza danych bądź pojedyncze pliki danych.
38.	Możliwość uruchomienia bazy danych w środowisku klastra wielu aktywnych serwerów bazy danych. Ilość socketów CPU w klastrze nie przekracza 2.
39.	Zwiększenie bądź zmniejszenie liczby serwerów obsługujących klastrową bazę danych nie może powodować konieczności reorganizacji fizycznej bazy danych (struktura plików danych).