Database Minimum Requirement Specification

**General**

1. Database should be available on standard operating systems like Windows, Linux ~~and UNIX.~~
2. The proposed database should conform to latest major version 6.0 or higher of Unicode with Indian Language support.
3. Database should work seamlessly with ODBC, JDBC, Java, .NET, PHP and other open standards.
4. All the features of database required should be integrated with database engine without any 3rd party products or overlay products.
5. Database should have capability of storing scanned images, text documents, xml, and multi-media inside the tables. It should be part of the basic database distribution without any additional cost to the organization.
6. Database should provide capability to monitor and manage the database using web based tool with additional functionality for performance management and database management.
7. Database should provide capability to monitor and administer the database, database clusters, monitor performance, maintain database, backup and recovery, disaster recovery management using web based tool with options to do diagnosis, performance tuning, finding the events, advisory based tuning mechanisms with the history.
8. Database should have an ability to monitor and self-tune its memory parameters dynamically when the executing workload changes
9. Database should allow the DBA to control size of memory allocations for buffer pool and should also recommend the optimal size of the buffer pool for the workload.
10. The proposed database should support all industry standard hardware platforms (CISC/ RISC/ EPIC) and operating systems.

**Scalability**

1. Database should provide horizontal scalability in such a manner that a new database node can be added (or removed) dynamically, as and when required in future, without disturbing the normal functioning of production system such as without shutdown. ~~This should be supported by native database clustering components~~

**Availability**

1. Database should provide continuous availability features to address hardware failures, instance failures, human errors like accidental deletion of data, tables, able to repair at block level. These features should be available on all the supported platforms. It should provide commands to:
   1. Retrieve historical data of a table for a specific time interval to view all the rows of a table that ever existed during a given time interval.
   2. Retrieve historical data for a given transaction, or for all transactions within a given time interval.
2. Database should have a Disaster- Recovery (DR) solution to replicate the changes happening in the database to the DR site both in synchronous and asynchronous mode and should have provision to configure setup for zero data loss
3. Database should have built-in DR solution to replicate the changes happening in the database across multiple DR Sites, with a provision to configure the setup for zero data loss.
4. Database should provide continuous availability features to address hardware failures, instance failures, human errors like accidental deletion of data, tables, able to repair at block level.
5. Database should have support to automatically extend storage as needed to a defined limit (or physical limit) thereby reducing unplanned outage at the database and table space levels
6. Database should provide the features whereby database objects (e.g. tables, indexes etc.) can be redefined while the production application is in use, thereby minimizing or eliminating down time.

**Performance**

1. Database should be able to provide database level storage management mechanism, which should enable the availability by means of creating redundancy, ~~automatically balance~~ the data files across the available disks, ~~I/O balancing across the available disks for the database~~ for performance, availability and management.
2. Database should support option of different partitioning schemes within the database (Range, List, Hash & combinations etc.) to split large volumes of data into separate pieces or partitions, which can be managed independently. ~~Partitioning based on physical and virtual table columns should be supported.~~
3. Database should have capability to suggest which columns should be candidate for partitioning.
4. Database should provide continuous availability features to address hardware failures, instance failures, human errors like accidental deletion of data, tables, able to repair at block level. ~~These features should be available on all the supported platforms.~~
5. Database should offer parallel query and indexing for performance of queries of very large vector datasets.
6. Database should support spatial data types that provide spatial object type storage ~~and it should support r-tree indexes for spatial objects~~.
7. Database should provide functionality of materialized views which are the database objects that contains the result of query that may be subset of the rows and/or columns of a table or join result. This is to provide efficient access to data. ~~The functionality should be provided without using the use of database triggers as that adversely effects the database performance.~~

**Security**

1. The database should provide controls over who, when, where and how applications, data and databases can be accessed.
2. Database should support the separation of security functionality from application functionality and database administration functionality.
3. Database should be possible to prevent privileged IT users such as DBAs and administrators from accessing and modifying the data.
4. The database should provide multi-factor authentication-based controls and policies taking account of IP addresses, application context etc.
5. Database should provide functionality to restrict the access to database tables through the application only. It should restrict users or DBA or any privileged user accessing the operational information through SQL Language / Tools like Toad etc., using direct connection.
6. A comprehensive security at Database level should be in place to prevent application bypass and also prevent access of department‘s data privileged IT users.
7. Audit trail should also be maintained at database level for any changes made in database and it should be ensured that these audit trails cannot be manipulated by anyone including super users and DBAs.
8. Solution should have a centrally managed secure audit storage system which is able to capture before and after values from transaction logs, privileged user audits, raise alters on suspicious activity. It should also provide security facilities for role segregation within auditing functions in terms of administration, auditors etc.
9. Audit system should enforce separation of duties between auditors and administrator.
10. Database should capture before / after values from transaction logs and raise alert on critical and suspicious activity.
11. The Database solution should audit stored procedure calls and also recursive Sql which runs within the database. It should provide user role audit (change in users role, privileges)
12. Along with storing audit logs, it should be possible to generate and send alters if some authorized access or update happens on critical data. The solution should provide built-in and customizable reports for department compliance and monitoring.
13. Network-based access control or monitoring solution should be provided which can be used to stop SQL injection/insertion attacks on database.
14. Database should support native optional database level encryption on the table columns, table spaces or backups. It should support for enhanced authentication by integrating tokens and biometric technologies.
15. It should support Table level encryption so that entire application tables can be transparently encrypted. Data blocks should be transparently decrypted as they are accessed by the database.
16. Database should provide strong encryption capabilities within database for stored information in the tables as well as the information transmitted over network.
17. Database should have capability to integrate with HSM and store the encryption keys in HSM.
18. The encryption process should be able to utilize CPU-based hardware acceleration of industry processors like Intel® AES-NI
19. Database should provide on-the-fly redaction of sensitive data in database query results prior to display by applications so that unauthorized users cannot view the sensitive data. The command level API and GUI should be made available to define the redaction policies and rules.
20. Policy based backup solution should be implemented and the backup should be encrypted whether stored on disk or tapes.
21. The proposed database in the solution should have received the security certification of level 4 (EAL4) or above from the International Common Criteria.

**Semantics Capability**

1. Database should have advanced semantic data management capability and should natively support storing, loading and DML operations on ontology and RDF/OWL models.
2. Database should support native inference engine for efficient and scalable inference using OWL semantics, RDF, RDFS, and user defined rules.
3. Database should be possible to query RDF/OWL data using Sql queries.
4. Database should provide fine-grained security where it should be possible to restrict user‘s access to triples that involve instances of a specific RDF class.
5. Database should support semantic indexing for documents that can locate and extract meaningful information from unstructured documents to semantically index documents stored in relational tables.
6. Database RDF and OWL data types should be compliant with open W3C standards so that the database can be interoperable knowledge base.
7. Database should support SQL, SPARQL and Ontology assisted query on semantic data.
8. Database should be able to utilize database security policies and data classification for RDF data.
9. Database should support Semantic indexing of documents based on popular natural language