**IMPLEMENTATION**

**MODULES:**

1. Setup Phase
2. Meta Data Module
3. Sequential SQL Operations
4. Concurrent SQL Operations

**MODULES DESCRIPTION:**

**Setup Phase:**

* We describe how to initialize a Secure DBaaS architecture from a cloud database service acquired by a tenant from a cloud provider.
* We assume that the DBA creates the metadata storage table that at the beginning contains just the database metadata, and not the table metadata.
* The DBA populates the database metadata through the Secure DBaaS client by using randomly generated encryption keys for any combinations of data types and encryption types, and stores them in the metadata storage table after encryption through the master key.
* Then, the DBA distributes the master key to the legitimate users. User access control policies are administrated by the DBA through some standard data control language as in any unencrypted database. In the following steps, the DBA creates the tables of the encrypted database.

**Meta Data Module:**

* In this module, we develop Meta data. So our system does not require a trusted broker or a trusted proxy because tenant data and metadata stored by the cloud database are always encrypted.
* In this module, we design such as Tenant data, data structures, and metadata must be encrypted before exiting from the client.
* The information managed by SecureDBaaS includes plaintext data, encrypted data, metadata, and encrypted metadata. Plaintext data consist of information that a tenant wants to store and process remotely in the cloud DBaaS.
* SecureDBaaS clients produce also a set of metadata consisting of information required to encrypt and decrypt data as well as other administration information. Even metadata are encrypted and stored in the cloud DBaaS.

**Sequential SQL Operations:**

* The first connection of the client with the cloud DBaaS is for authentication purposes. Secure DBaaS relies on standard authentication and authorization mechanisms pro-vided by the original DBMS server. After the authentication, a user interacts with the cloud database through the Secure DBaaS client.
* Secure DBaaS analyzes the original operation to identify which tables are involved and to retrieve their metadata from the cloud database. The metadata are decrypted through the master key and their information is used to translate the original plain SQL into a query that operates on the encrypted database.
* Translated operations contain neither plaintext database (table and column names) nor plaintext tenant data. Nevertheless, they are valid SQL operations that the Secure DBaaS client can issue to the cloud database. Translated operations are then executed by the cloud database over the encrypted tenant data. As there is a one-to-one correspondence between plaintext tables and encrypted tables, it is possible to prevent a trusted database user from accessing or modifying some tenant data by granting limited privileges on some tables.
* User privileges can be managed directly by the untrusted and encrypted cloud database. The results of the translated query that includes encrypted tenant data and metadata are received by the Secure DBaaS client, decrypted, and delivered to the user. The complexity of the translation process depends on the type of SQL statement.

**Concurrent SQL Operations:**

* The support to concurrent execution of SQL statements issued by multiple independent (and possibly geographically distributed) clients is one of the most important benefits of Secure DBaaS with respect to state-of-the-art solutions.
* Our architecture must guarantee consistency among encrypted tenant data and encrypted metadata because corrupted or out-of-date metadata would prevent clients from decoding encrypted tenant data resulting in permanent data losses.
* A thorough analysis of the possible issues and solutions related to concurrent SQL operations on encrypted tenant data. Here, we remark the importance of distinguishing two classes of statements that are supported by Secure DBaaS: SQL operations not causing modifications to the database structure, such as read, write, and update; operations involving alterations of the database structure through creation, removal, and modification of database tables (data definition layer operators).