# Database Strategy for MTA

## FIS Demands

* Support multiple tenants
* Each tenant will have individual look & feel (branding)
* Each tenant will have internal business workflows in addition to certain generic ones
* System needs to have defense in depth security
* Data structure should be customizable on per-tenant basis
* Database structure should have minimum maintenance overhead

## Deployment Packaging

Base deployment package will contain a website configuration and tenant configuration.

The website configuration will contain details of various available components like menu, carousal. The site admin can configure how these components will be displayed on the end user’s frontend site.

The tenant configuration will have global details for each tenant serviced by FIS and master data concerning the business domain itself (like various card types available). Along with this, it will have a base schema containing master (end-customer specific) and transactional (card load and payment specific) tables.

## Tenant Provisioning

The website configuration will be contained in a single database instance for all tenants serviced. The tenant configuration will be initiated in main database upon provisioning.

Upon provisioning a new tenant in the system, a replica of site’s configuration will be created in the site configuration database. This will have the exact look-and-feel of base FIS website. A copy of tenant specific transactional data-objects (concerning their own customers) will be created in tenant specific schema (logical partition) in the main database.

## Security

The site configuration will be only available to tenant’s website administrators. Any global configuration can be applied by FIS’s own administrators. The tenant will be given a dedicated user through which its data will be accessed, and the ownership will be within that user’s own scope. This way, no other entity can access other’s data, eliminating security concerns.

## Scalability and Performance

Each tenant will have its own set of scaling and performance requirements, based on user load. Hence the base system will be offered to be deployed in an “in-premise” database system for start. Later, as load grows, the entire logical partition can be moved to a physical database by just moving a particular schema.

The system will offer multiple choices of target location for tenant DB.

1. Within same instance, as logical schema (default option)
2. Within same instance, different database
3. Within same server, different instances
4. Within same premise, different servers
5. Different premises
6. On cloud

## DB Structure

### Domain data

These will be in main database, shared for all clients. Example tables as given

* Card Types (master)
* License / SLA
* Tenant

### Tenant Specific

These will be contained in a logical schema and physical deployment will be the choice of the tenant concerned. Example tables as given:-

* Customer (end user)
* Transaction (each transaction record)
* Audit

Any procedure designed to maintain such data will also be on individual basis for tenant.

Upon provisioning of tenant, following will be created:-

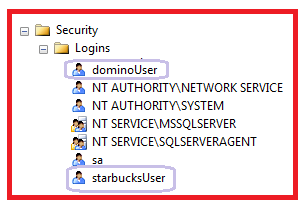
* Tenant entry (starbucks/dominos) and its corresponding license/SLA details
* Database user for administrative and user level access
* Database schema under which all transactional objects and logic (stored procedure/views) lie.
* Copy of base transactional schema, comprising different transactional (end-customer related) tables and sql components (SP/function/views) to manipulate those.

## DB Design Structure

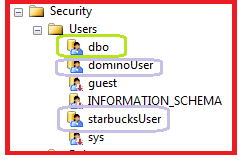
Legend

* Main tables: Light green
* Base schema for transaction tables: light brown
* Replicated tables for specific tenants: Violet

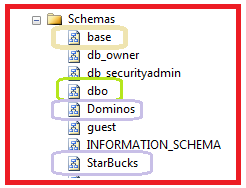
### DB Logins



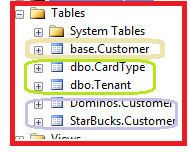
### DB Users



### Schemas



### Tables



### Stored Procedures

