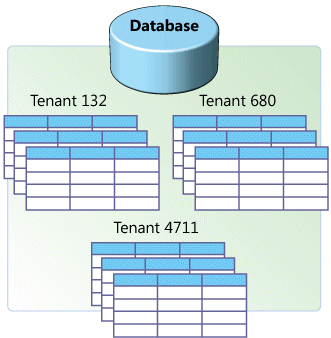
**Multi-tenant**

Esta arquitectura de software se agregó a la versión de Oracle 12c que permite a una sola instancia de software servir a muchos clientes. Donde, un solo desarrollo de código puede servir para multiples usuarios, donde se separa la información sensible de cada uno y que solo sea visible por ellos.

En esta arquitectura permite que se pueda personalizar elementos de la aplicación, como son colores de la interfaz, pero no se personaliza el código como tal. Donde lo mas importante de esta arquitectura es que un cliente no necesariamente es un usuario único, puede ser un grupo de usuarios. Por ejemplo algunos servicios funcionan los teams o grupos en los que un cliente puede tener un subdominio para entrar a la aplicación y este cliente puede tener multiples usuarios que ven la información de ese cliente.



**Ventajas y desventajas de un multi-tentant**

* *Ventajas.*

Economía en desarrollo y mantenimiento, ya que los costos son distribuidos entre todos los clientes.

Fácil actualización, ya que es necesario solo actualizar una sola instancia.

Seguridad de la información de cada cliente, ya que cuenta con un schema separado para cada uno.

Optimiza el uso de recursos de los servidores.

* *Desventajas*

Dificulta el desarrollo de características específicas para un cliente.

Único punto de falla: si la aplicación tiene un error o falla, fallará para todos los clientes.

**Arquitectura Multi-tenant aplicada a bases de datos.**

Existen tres tipos principales de arquitecturas

1. **Compartida (Shared)**

*Una Base de Datos - Un Schema*

Esta arquitectura es la que se usa por defecto en la mayoría de aplicaciones, se separa los usuarios de la aplicación por una relación con la tabla de usuarios. Es útil para servicios B2C como Evernote o LastPass en sus planes individuales.

1. **Separada (Isolated)**

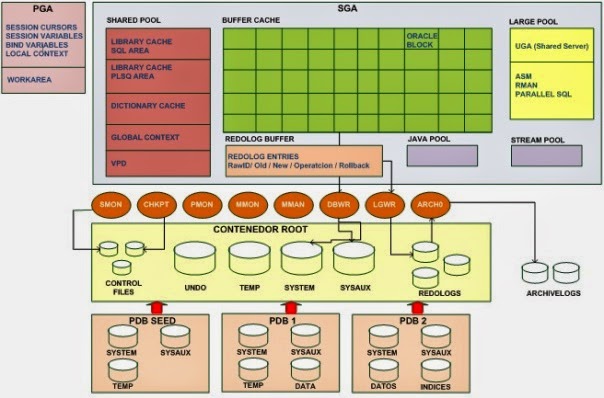
*Cada cliente (tenant) tiene su propia base de datos*

Esta arquitectura permite usar cualquier motor de base de datos, se puede relacionar como el proceso de virtualización en el que cada instancia se implementa para el nuevo cliente.

1. **Híbrida (Semi - Isolated)**

*Una base de datos - Multiples Schemas*

Esta arquitectura usa lo mejor de las dos anteriores y es posible usarla en PostgreSQL. Esto permite, por ejemplo, que cada uno de los clientes tenga usuarios y sus datos estén separados por cada schema.



*Nueva arquitectura multitenat Oracle 12c*