**Proyecto del curso – Iteración 3**

John Alexander Guzmán Moyano, Mario Felipe Medina Zamudio

Universidad de los Andes, Bogotá, Colombia

{ja.guzmanm, mf.medinaz}@uniandes.edu.co

Fecha de presentación: Noviembre 16 de 2020

**Tabla de contenido**

[1 Revisión Caso de Estudio 1](#_Toc54036190)

[1.1 Roles de Usuario 1](#_Toc54036191)

[1.2 Entidades de Negocio 1](#_Toc54036192)

[1.3 Funcionalidades Principales 1](#_Toc54036193)

[1.4 Reglas de Negocio 1](#_Toc54036194)

[2 Análisis y Modelo Conceptual 2](#_Toc54036195)

[2.1 Requerimientos Funcionales 2](#_Toc54036196)

[3 Diseño de la base de datos 9](#_Toc54036197)

[3.1 Modelo de datos relacional 9](#_Toc54036198)

[3.2 Normalización 10](#_Toc54036199)

[4 Control de calidad del modelo 10](#_Toc54036200)

[4.1 Verificación Requerimientos Funcionales 10](#_Toc54036201)

[4.2 Verificación Requerimientos No Funcionales 15](#_Toc54036202)

[5 Resultados logrados 16](#_Toc54036203)

# Cambios Al Modelo Relacional

## Adición de Estados

Nuestro modelo relacional recibió la adición de los posibles estados de los visitantes y los espacios, como se puede ver a continuación (también está el archivo de Excel en la carpeta doc):



## Normalización

**1FN**: Se encuentra en esta forma ya que en el modelo de datos no se tiene ningún atributo multivalor (todos los atributos tienen valores atómicos).

**2FN**: Está en esta forma normal debido a que no existen dependencias parciales desde el atributo primo (en nuestro caso solo hay un atributo primo y es el ID de cada clase).

**3FN:** Como ningún atributo no primo está determinado por otro atributo no primo; los atributos no primos se definen de forma no transitiva por la llave candidata, no existen dependencias transitivas entre atributos no primos. Esto implica que el modelo de datos se encuentra en esta forma normal.

**BCNF**: Como las llaves de las tablas son únicamente el ID de la clase, significa que esta es simple, de modo que el modelo está en esta forma normal.

## Verificación Requerimientos No Funcionales

* RNF1 – Privacidad:

Se facilita ya que el modelo solo permite a los visitantes consultar la información acerca de sus propias visitas, por lo que ningún otro usuario puede hacer uso de esa información. El único que tiene permitido consultar información de otros es el administrador

* RNF2 – Persistencia:

Se facilita porque la información se persiste en una base de datos y está distribuida de forma que cuando un requerimiento funcional necesita información acerca de una entidad puede acceder a esta en específico, ahorrando uso de memoria principal.

* RNF3 – Concurrencia:

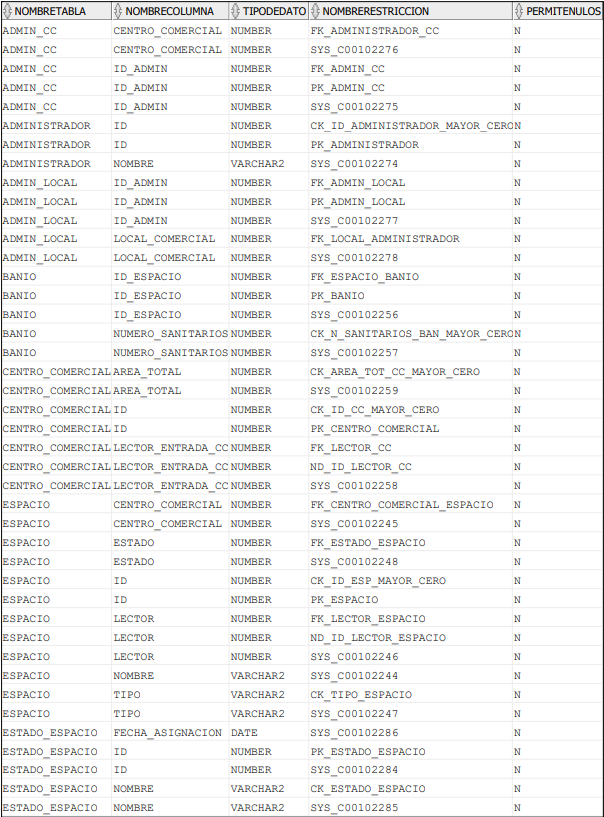
Se facilita porque nuestro sistema implementaría un SMBD. Con la ayuda del SMBD nuestro modelo separa las entidades de forma que los requerimientos funcionales se pueden llevar acabo de forma independiente.

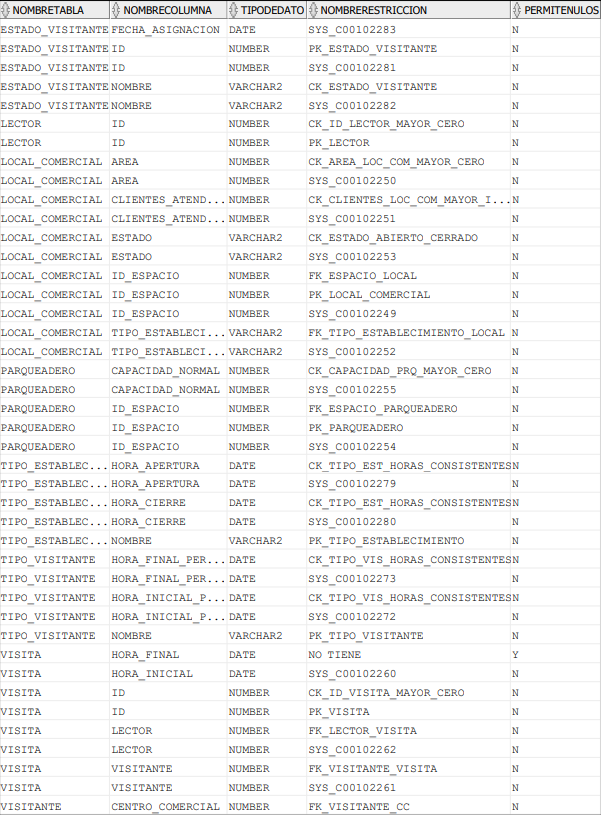
* RNF4 – Distribución:

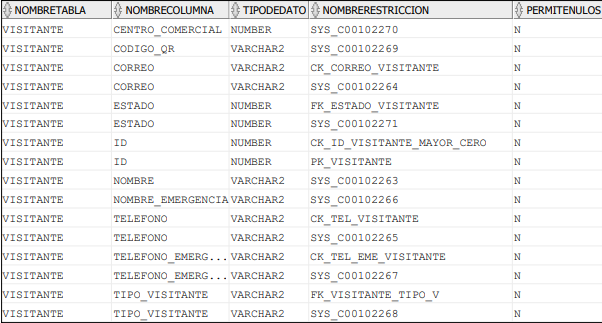
Todo el modelo está pensado para trabajar sobre una base de datos que maneja la información de todo el programa sin importar la entidad que se quiera guardar.

## Listado con las tablas generadas en la base de datos

El sql de las siguientes listas se encuentra en la carpeta doc como “Listatablas.sql”







# Resultados logrados

Realizamos los nuevos requerimientos funcionales y requerimientos funcionales de consulta, que están implementados en la aplicación (en la interfaz app), a excepción del RFC7. No hay escenarios de prueba.