“Transacciones distribuidas”

Juan Sebastián Díaz Serrano, Jorge Gómez, Sergio Guzmán Mayorga, Mauricio Neira Felipe Ramos, Juan Camilo Ruiz

Iteración 5

Universidad de los Andes, Bogotá, Colombia

{js.diaz, ja.gomez10, s.guzmanm, m.neira10, jf.ramos, jc.ruiz[}@uniandes.edu.co](mailto:%7d@uniandes.edu.co)

Fecha de presentación: Diciembre 10 de 2017

Tabla de contenido

[1 Introducción 1](#_Toc494913959)

[2 Modelos utilizados 1](#_Toc494913960)

[3 Distribución del trabajo 3](#_Toc494913961)

[4 Requerimientos funcionales 4](#_Toc494913962)

[4.1 RF9 4](#_Toc494913963)

[4.2 RF10 4](#_Toc494913964)

[5 Requerimientos de consulta 4](#_Toc494913965)

[5.1 RFC1 4](#_Toc494913966)

[5.2 RFC2 4](#_Toc494913967)

[5.3 RFC3 5](#_Toc494913968)

[6 Consideraciones adicionales 5](#_Toc494913969)

[7 Análisis de resultados 6](#_Toc494913970)

[7.1 Aprendizajes y logros 6](#_Toc494913971)

[7.2 Conclusiones 6](#_Toc494913972)

[8 Bibliografía 6](#_Toc494913973)

# Análisis de la implementación de transacciones distribuidas

## Restricciones de rotondas

A continuación se presentan las restricciones planteadas de cada rotonda para su negocio. Se asume que todas las rotondas del sistema utilizan una interfaz *XA* de *X/Open* junto con la implementación de *RabbitMQ* para el manejo de colas de mensajes

### Rotonda A-05

* Para que un restaurante esté registrado en la rotonda debe tener un representante como usuario de la rotonda. Dicho representante presenta un rol de LOCAL.
* Se asume que hay restaurantes que pueden ofrecer el mismo producto al usuario. Sin embargo, lo referente a disponibilidad, precios y costos dependen de cada local.
* Pueden existir dos menús con el mismo nombre en la rotonda, pero la descripción, costo, productos, precio, etc, dependen del restaurante que lo está ofreciendo.
* Una cuenta tiene o no asignada un cliente y a su vez puede o no estar registrada a una mesa. Ahora por la conformación de la “superrotonda” se plantea que la mesa sea representada por una cadena de caracteres de la forma:

<Número de la mesa>-<Nombre de la zona>-<Nombre de la rotonda>

* Todo producto debe tener asignado uno de los 5 tipos definidos en la rotonda: PLATO FUERTE, ACOMPAÑAMIENTO, BEBIDA, POSTRE y ENTRADA.
* Ningún producto o ingrediente puede ser sustituido por sí mismo.
* Ningún producto o ingrediente puede ser sustituto por otro ítem ofrecido en un restaurante diferente a donde proviene.
* El número de cuenta es generado automáticamente por el sistema.
* Los roles posibles de un usuario registrado son: CLIENTE, ORGANIZADOR, OPERADOR, LOCAL y PROVEEDOR.
* No hay menús que posean 2 platos del mismo tipo en el sistema.
* Un usuario puede presentar preferencias por precios, categorías o zonas que pertenecen únicamente al sistema local de la rotonda (No se tiene en cuenta las alianzas que haya con otras rotondas).
* Las reservas solo se pueden realizar en la rotonda local.

### Rotonda A-17

### Rotonda B-14

## Ajuste de arquitectura

A continuación, se presenta la forma en que quedaron los modelos relacionales de cada subgrupo:

## Lógica del requerimiento RF18

## Estrategias para el cumplimiento de casos de uso

### Colas de mensajes

### *Two-Phase Commit*

### Comparación

# Especificación e implementación de transacciones distribucciones distribuidas (Análisis de impacto de estrategias)

## RF18

## RF19

## RFC13

## RFC14

# Consideraciones adicionales

Dentro de la carpeta data se tienen las siguientes consideraciones:

* Se tienen los archivos de pruebas de *Postman* en la carpeta pruebas. Antes de ejecutarlas se debe correr el archivo PrePostman.sql.
* En el archivo BorradoEInserciones.sql se tienen los comandos para borrar todos los datos existentes en la base de datos y volver a poblarla en caso tal de que se corrompan de laguna manera los datos del sistema.
* En el archivo DBSETUPFULL.sql se tienen los esquemas de las tablas para reconstruir la base de datos dado el caso en que halla necesidad de hacerlo.
* En la carpeta inicializacion están todas las inserciones en tablas utilizadas en la población de la base de datos.
* La ruta se sigue llamando VideoAndes de acuerdo con el ejemplo de clase, dado que no se encontró la forma de cambiar el nombre a RotondAndes como se pretendía.

# Análisis de resultados

## Aprendizajes y logros

A lo largo del desarrollo de la iteración 2 se aprendió a crear un sistema transaccional básico con múltiples iteraciones entre clases que realiza algunas consultas interesantes.

## Conclusiones

* Se debe saber determinar las operaciones transaccionales del sistema porque hay varias que requieren ser o todo o nada.
* SQL es una herramienta poderosa que permite realizar consultas complejas sobre modelos de datos gigantes.
* La interconexión entre cñases permite la funcionalidad completa de un sistema transaccional dado.
* Toda decisión de diseño se debe reflejar en la base de datos planteada.

# Bibliografía

1. Universidad de los Andes. *Taller catálogo y SQL básico.* [En línea] Universidad de los Andes. [Citado el: 4 de Octubre de 2017.] https://sicuaplus.uniandes.edu.co/bbcswebdav/pid-1918111-dt-content-rid-20097375\_1/courses/UN\_201720\_ISIS2304/isis2304-172-CasoEstudio-RotondAndes-v2.pdf