

**ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO**

**FACULTAD DE INFORMÁTICA Y ELECTRÓNICA**

**ESCUELA DE INGENIERÍA EN SISTEMAS**

**ANÁLISIS COMPARATIVO ENTRE LOS RDBMS LIGEROS SQLITE Y SQL SERVER COMPACT EN WINDOWS PHONE PARA EL SISTEMA DE FACTURACIÓN DEL HOTEL LOS LIRIOS.**

Tesis de grado para optar por el grado académico de:

**INGENIERO EN SISTEMAS INFORMÁTICOS**

**AUTORES:** DARWIN VLADIMIR CEDEÑO BETANCOURT

DIEGO MAURICIO QUINATOA CANDO

**TUTOR:** ING. JORGE MENÉNDEZ VERDECIA

**Riobamba – Ecuador**

**-2016-**

**ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO**

**FACULTAD DE INFORMÁTICA Y ELECTRÓNICA**

**ESCUELA DE INGENIERÍA EN SISTEMAS**

El tribunal de tesis certifica que: El trabajo de investigación: “ANÁLISIS COMPARATIVO ENTRE LOS RDBMS LIGEROS SQLITE Y SQL SERVER COMPACT EN WINDOWS PHONE PARA EL SISTEMA DE FACTURACIÓN DEL HOTEL LOS LIRIOS”*,* de responsabilidad de los señores DIEGO MAURICIO QUINATOA CANDO y DARWIN VLADIMIR CEDEÑO BETANCOURT, ha sido minuciosamente revisado por los miembros del tribunal de tesis, quedando autorizada su presentación.

|  |  |  |
| --- | --- | --- |
| **NOMBRES** | **FIRMAS** | **FECHA** |
| Dr. Miguel Tasambay Ph.D.  **DECANO DE LA FACULTAD DE INFORMÁTICA Y ELECTRÓNICA** | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Ing. Julio Santillán  **DIRECTOR DE LA ESCUELA DE INGENIERÍA EN SISTEMAS** | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Ing. Jorge Menéndez  **DIRECTOR DE TESIS** | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Ing. Germania Veloz  **MIEMBRO DE TRIBUNAL** | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

Nosotros, Diego Mauricio Quinatoa Cando y Darwin Vladimir Cedeño Betancourt somos los responsables de las ideas, doctrinas y resultados expuestos en esta tesis y el patrimonio intelectual de la misma pertenecen a la Escuela Superior Politécnica de Chimborazo.

|  |
| --- |
| **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** |
| Diego Mauricio Quinatoa Cando |
|  |
|  |
| **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** |
| Darwin Vladimir Cedeño Betancourt |

# AGRADECIMIENTO

Agradezco a Dios por darme la vida y darme unos buenos padres, a mi querida mamita que siempre estuvo al pendiente de mis estudios, jamás descuidó de mí siempre apoyándome con sus sabios consejos, a mi padre que está en el cielo que me enseño que la humildad y la sencillez te hacen una buena persona, también a mi querido hijo que me impulsó a terminar este objetivo planteado años atrás.

**Diego.**

Con estas sencillas palabras quiero agradecer principalmente a mis padres Luis y Mirian por darme la posibilidad de tener una formación académica y ser alguien en la vida, a mis hermanos, a mi gran amigo José Manuel Espinoza por su apoyo incondicional y en especial a mi esposa Elisa Schiavi y mi hijo Matías Gioele porque por ellos va es te logro, gracias a todos ellos por su confianza, tiempo y paciencia depositada en mí día a día.

**Darwin.**

TABLA DE CONTENIDO

[AGRADECIMIENTO iv](#_Toc445823069)

[RESUMEN xii](#_Toc445823070)

[SUMARY xiii](#_Toc445823071)

[INTRODUCCIÓN - 1 -](#_Toc445823072)

[Justificación - 2 -](#_Toc445823073)

[Justificación teórica*.* - 2 -](#_Toc445823074)

[Justificación Metodológica. - 3 -](#_Toc445823075)

[Justificación Práctica. - 3 -](#_Toc445823076)

[Objetivos - 5 -](#_Toc445823077)

[Objetivo general - 5 -](#_Toc445823078)

[Objetivos específicos - 5 -](#_Toc445823079)

[Hipótesis - 5 -](#_Toc445823080)

[CAPÍTULO I - 6 -](#_Toc445823081)

[1 MARCO TEÓRICO - 6 -](#_Toc445823082)

[1.1 Windows Phone - 6 -](#_Toc445823083)

[1.1.1 Historia - 6 -](#_Toc445823084)

[1.1.2 Versiones de Windows Phone - 7 -](#_Toc445823085)

[1.1.3 Características principales de Windows Phone - 8 -](#_Toc445823086)

[1.1.4 SDK (Software Development Kit - Kit de desarrollo de software). - 9 -](#_Toc445823087)

[1.1.5 Lenguajes de programación - 9 -](#_Toc445823088)

[1.1.6 Silverlight para Windows Phone - 10 -](#_Toc445823089)

[1.1.7 Windows Phone - Universal Apps - 10 -](#_Toc445823090)

[1.2 Sistemas de Gestión de Bases de Datos Relacionales - 11 -](#_Toc445823091)

[1.2.1 Objetivos de los RDBMS - 11 -](#_Toc445823092)

[1.2.2 RDBMS tipo compacto - 12 -](#_Toc445823093)

[1.3 SQL Server Compact Edition para Windows Phone - 12 -](#_Toc445823094)

[1.3.1 Historia - 13 -](#_Toc445823095)

[1.3.2 Características - 13 -](#_Toc445823096)

[1.3.3 Lenguaje de consulta - 14 -](#_Toc445823097)

[1.3.4 Arquitectura de SQL Server CE - 14 -](#_Toc445823098)

[1.3.5 Ventajas y Limitaciones de SQL Server Compact Edition - 15 -](#_Toc445823099)

[1.3.6 Tipos de datos soportados por SQL Server CE - 16 -](#_Toc445823100)

[1.3.7 Instalación de SQL Server CE. - 17 -](#_Toc445823101)

[1.4 SQLite para Windows Phone - 17 -](#_Toc445823102)

[1.4.1 Historia - 18 -](#_Toc445823103)

[1.4.2 Características de SQLite - 18 -](#_Toc445823104)

[1.4.3 Lenguaje de consulta para SQLite - 19 -](#_Toc445823105)

[1.4.4 Arquitectura de SQLite - 20 -](#_Toc445823106)

[1.4.5 Ventajas y Limitaciones de SQLite - 21 -](#_Toc445823107)

[1.4.6 Tipos de Datos soportados - 22 -](#_Toc445823108)

[1.4.7 Instalación - 23 -](#_Toc445823109)

[1.5 Metodologías Ágiles - 23 -](#_Toc445823110)

[1.5.1 Metodología SCRUM - 24 -](#_Toc445823111)

[CAPÍTULO II - 27 -](#_Toc445823112)

[2 MARCO METODOLÓGICO - 27 -](#_Toc445823113)

[2.1 Tipo de Investigación - 27 -](#_Toc445823115)

[2.2 Parámetros de comparación - 27 -](#_Toc445823116)

[2.3 Ponderación de los parámetros de comparación - 28 -](#_Toc445823117)

[2.4 Diseño del experimento - 29 -](#_Toc445823118)

[2.4.1 Cálculo de la muestra - 30 -](#_Toc445823119)

[2.4.2 Descripción del ambiente de pruebas - 30 -](#_Toc445823120)

[2.4.3 Prototipos para las pruebas - 31 -](#_Toc445823121)

[CAPÍTULO III - 33 -](#_Toc445823122)

[3 RESULTADOS, DISCUSIÓN Y ANÁLISIS Y DE RESULTADOS - 33 -](#_Toc445823123)

[3.1 Análisis preliminar de datos - 33 -](#_Toc445823124)

[3.2 Estudio Comparativo - 36 -](#_Toc445823125)

[3.2.1 Eficiencia en Insert - 37 -](#_Toc445823126)

[3.2.2 Eficiencia en Update - 47 -](#_Toc445823127)

[3.2.3 Eficiencia en Delete - 57 -](#_Toc445823128)

[3.2.4 Eficiencia en Select - 67 -](#_Toc445823129)

[3.3 Resultados generales - 78 -](#_Toc445823130)

[3.4 Demostración de hipótesis - 79 -](#_Toc445823131)

[CAPÍTULO IV - 81 -](#_Toc445823132)

[4 SISTEMA DE FACTURACIÓN Y ADMINISTRACIÓN DEL HOTEL “LOS LIRIOS” (Lirios Fadmin) - 81 -](#_Toc445823133)

[4.1 Hotel “Los Lirios” panorama general - 81 -](#_Toc445823134)

[4.2 Roles SCRUM - 82 -](#_Toc445823135)

[4.3 Planificación - 82 -](#_Toc445823136)

[4.3.1 Product Backlog - 82 -](#_Toc445823137)

[4.3.2 Planificación de entregables. - 87 -](#_Toc445823138)

[4.4 Desarrollo - 89 -](#_Toc445823139)

[CONCLUSIONES - 101 -](#_Toc445823140)

[RECOMENDACIONES - 102 -](#_Toc445823141)

**ÍNDICE DE TABLAS**

[Tabla 1‑1: Tipos de datos para SQL Server CE. - 16 -](#_Toc445823151)

[Tabla 1‑2: Tipos de datos SQLite. - 22 -](#_Toc445823152)

[Tabla 2‑1: Parámetros de comparación. - 28 -](#_Toc445823153)

[Tabla 2‑2: Ponderación de los parámetros - 29 -](#_Toc445823154)

[Tabla 2‑3: Especificación hardware del equipo - 31 -](#_Toc445823155)

[Tabla 2‑4: Prototipos para las pruebas - 31 -](#_Toc445823156)

[Tabla 2‑5: Cantidad de datos obtenidos prototipos SQLite - 31 -](#_Toc445823157)

[Tabla 2‑6: Cantidad de datos obtenidos prototipos SQL Server Compact Edition - 32 -](#_Toc445823158)

[Tabla 3‑1: Prueba de normalidad prototipo Insert - 34 -](#_Toc445823159)

[Tabla 3‑2: Prueba de normalidad prototipo Update - 34 -](#_Toc445823160)

[Tabla 3‑3: Prueba de normalidad prototipo Delete - 35 -](#_Toc445823161)

[Tabla 3‑4: Prueba de normalidad prototipo Select. - 36 -](#_Toc445823162)

[Tabla 3‑5: Estadística descriptiva del uso CPU unidades de tiempo - Insert - 37 -](#_Toc445823163)

[Tabla 3‑6: Estadísticos de contraste uso CPU unidades de tiempo - prototipo Insert - 38 -](#_Toc445823164)

[Tabla 3‑7: Ponderación parámetro CPU unidades de tiempo - prototipo Insert - 39 -](#_Toc445823165)

[Tabla 3‑8: Estadística Descriptiva uso CPU unidades de procesamiento - prototipo Insert - 40 -](#_Toc445823166)

[Tabla 3‑9: Estadísticos de contraste uso CPU unidades de procesamiento - prototipo Insert - 41 -](#_Toc445823167)

[Tabla 3‑10: Ponderación parámetro uso CPU unidades de procesamiento - prototipo Insert - 42 -](#_Toc445823168)

[Tabla 3‑11: Estadística Descriptiva uso RAM - prototipo Insert - 43 -](#_Toc445823169)

[Tabla 3‑12: Estadísticos de contraste uso RAM - prototipo Insert - 44 -](#_Toc445823170)

[Tabla 3‑13: Ponderación parámetro uso RAM - prototipo Insert - 44 -](#_Toc445823171)

[Tabla 3‑14: Tamaños de archivo Almacenamiento - prototipo Insert - 45 -](#_Toc445823172)

[Tabla 3‑15: Ponderación parámetro uso Almacenamiento - prototipo Insert - 46 -](#_Toc445823173)

[Tabla 3‑16: Estadística Descriptiva uso CPU unidades de tiempo - prototipo Update - 47 -](#_Toc445823174)

[Tabla 3‑17: Estadísticos de contraste uso CPU unidades de tiempo - prototipo Update - 48 -](#_Toc445823175)

[Tabla 3‑18: Ponderación parámetro CPU unidades de tiempo - Update - 49 -](#_Toc445823176)

[Tabla 3‑19: Estadística Descriptiva uso CPU unidades de procesamiento - prototipo Update - 50 -](#_Toc445823177)

[Tabla 3‑20: Estadísticos de contraste uso CPU unidades de procesamiento - prototipo Update - 51 -](#_Toc445823178)

[Tabla 3‑21: Ponderación parámetro uso CPU unidades de procesamiento - Update - 51 -](#_Toc445823179)

[Tabla 3‑22: Estadística Descriptiva uso RAM - prototipo Update - 53 -](#_Toc445823180)

[Tabla 3‑23: Estadísticos de contraste uso RAM - prototipo Update - 54 -](#_Toc445823181)

[Tabla 3‑24: Ponderación parámetro uso RAM - prototipo Update - 54 -](#_Toc445823182)

[Tabla 3‑25: Tamaños de archivo Almacenamiento - prototipo Update - 56 -](#_Toc445823183)

[Tabla 3‑26: Ponderación parámetro uso Almacenamiento - prototipo Update - 56 -](#_Toc445823184)

[Tabla 3‑27: Estadística Descriptiva uso CPU unidades de tiempo - prototipo Delete - 57 -](#_Toc445823185)

[Tabla 3‑28: Estadísticos de contraste uso CPU unidades de tiempo - prototipo Delete - 58 -](#_Toc445823186)

[Tabla 3‑29: Ponderación parámetro CPU unidades de tiempo - prototipo Delete - 59 -](#_Toc445823187)

[Tabla 3‑30: Estadística descriptiva uso CPU unidades de procesamiento - prototipo Delete - 60 -](#_Toc445823188)

[Tabla 3‑31: Estadísticos de contraste uso CPU unidades de procesamiento - prototipo Delete - 61 -](#_Toc445823189)

[Tabla 3‑32: Ponderación parámetro CPU unidades de tiempo - prototipo Delete - 61 -](#_Toc445823190)

[Tabla 3‑33: Estadística descriptiva uso RAM - prototipo Delete - 63 -](#_Toc445823191)

[Tabla 3‑34: Estadísticos de contraste uso RAM - prototipo Delete - 64 -](#_Toc445823192)

[Tabla 3‑35: Ponderación parámetro uso RAM - prototipo Delete - 64 -](#_Toc445823193)

[Tabla 3‑36: Tamaños de archivo Almacenamiento - prototipo Delete - 65 -](#_Toc445823194)

[Tabla 3‑37: Ponderación parámetro uso Almacenamiento – prototipo Delete - 66 -](#_Toc445823195)

[Tabla 3‑38: Estadística Descriptiva uso CPU unidades de tiempo - prototipo Select - 67 -](#_Toc445823196)

[Tabla 3‑39: Estadísticos de contraste uso CPU unidades de tiempo - prototipo Select - 68 -](#_Toc445823197)

[Tabla 3‑40: Ponderación parámetro uso CPU unidades de tiempo - prototipo Select - 69 -](#_Toc445823198)

[Tabla 3‑41: Estadística Descriptiva uso CPU unidades de procesamiento - prototipo Select - 70 -](#_Toc445823199)

[Tabla 3‑42: Estadísticos de contraste uso CPU unidades de procesamiento – prototipo Select - 71 -](#_Toc445823200)

[Tabla 3‑43: Ponderación parámetro uso CPU unidades de procesamiento - prototipo Select - 71 -](#_Toc445823201)

[Tabla 3‑44: Estadística Descriptiva uso RAM – prototipo Select - 73 -](#_Toc445823202)

[Tabla 3‑45: Estadísticos de contraste uso RAM – prototipo Select - 74 -](#_Toc445823203)

[Tabla 3‑46: Ponderación parámetro uso RAM – prototipo Select - 74 -](#_Toc445823204)

[Tabla 3‑47: Tamaños de archivo Almacenamiento – prototipo Select - 75 -](#_Toc445823205)

[Tabla 3‑48: Ponderación parámetro uso Almacenamiento – prototipo Select - 75 -](#_Toc445823206)

[Tabla 3‑49: Resultado de las medias de uso de los recursos hardware obtenidas con la estadística descriptiva para el RDBMS SQLite - 76 -](#_Toc445823207)

[Tabla 3‑50: Resultado de las medias de uso de los recursos hardware obtenidas con la estadística descriptiva para el RDBMS SQL Server Compact - 77 -](#_Toc445823208)

[Tabla 3‑51: Resultados generales - 78 -](#_Toc445823209)

[Tabla 4‑1: Product Backlog LiriosFadmin - 83 -](#_Toc445823210)

[Tabla 4‑2: Sprint Backlog LiriosFadmin. - 87 -](#_Toc445823211)

[Tabla 4‑3: Formato tarjetas de historia de usuario. - 89 -](#_Toc445823212)

[Tabla 4‑4: Modelo de prueba de aceptación. - 90 -](#_Toc445823213)

[Tabla 4‑5: Sprint Backlog–Sprint 1- Diseño de la solución, LiriosFadmin - 90 -](#_Toc445823214)

[Tabla 4‑6: Sprint Backlog–Sprint 2- Lirios Desktop - 98 -](#_Toc445823215)

[Tabla 4‑7: Sprint Backlog–Sprint 3 - Lirios Desktop - 98 -](#_Toc445823216)

[Tabla 4‑8: Sprint Backlog–Sprint 4 Lirios Mobile - Lirios Desktop - 99 -](#_Toc445823217)

**ÍNDICE DE ILUSTRACIONES**

[Figura 1‑1: Universal apps Windows Platform - 11 -](#_Toc445823218)

[Figura 1‑2: Arquitectura SQL Server CE - 15 -](#_Toc445823219)

[Figura 1‑3: Sentencias SQL - 19 -](#_Toc445823220)

[Figura 1‑4: Diagrama de bloques – Arquitectura SQLite - 20 -](#_Toc445823221)

[Figura 1‑5: Procesos de la metodología SCRUM - 26 -](#_Toc445823222)

[Figura 3‑1: Promedio de uso CPU unidades de tiempo (ms) - prototipo Insert - 39 -](#_Toc445823223)

[Figura 3‑2: Uso CPU unidades de procesamiento - prototipo Insert - 42 -](#_Toc445823224)

[Figura 3‑3: Promedio en bytes del uso del recurso RAM - prototipo Insert - 45 -](#_Toc445823225)

[Figura 3‑4: Promedio del uso Almacenamiento - prototipo Insert - 46 -](#_Toc445823226)

[Figura 3‑5: Promedio de uso CPU unidades de tiempo (ms) - prototipo Update - 49 -](#_Toc445823227)

[Figura 3‑6: Promedio de uso CPU unidades de procesamiento - Update - 52 -](#_Toc445823228)

[Figura 3‑7: Promedio en bytes del uso del recurso RAM - prototipo Update - 55 -](#_Toc445823229)

[Figura 3‑8: Promedio del uso Almacenamiento - prototipo Update - 56 -](#_Toc445823230)

[Figura 3‑9: Promedio de uso CPU unidades de tiempo (ms) – prototipo Delete - 59 -](#_Toc445823231)

[Figura 3‑10: Uso CPU unidades de procesamiento - prototipo Delete - 62 -](#_Toc445823232)

[Figura 3‑11: Promedio en bytes del uso del recurso RAM – prototipo Delete - 65 -](#_Toc445823233)

[Figura 3‑12: Promedio del uso Almacenamiento – prototipo Delete - 66 -](#_Toc445823234)

[Figura 3‑13: Promedio de uso CPU unidades de tiempo (ms) - prototipo Select - 69 -](#_Toc445823235)

[Figura 3‑14: Uso CPU unidades de procesamiento – prototipo Select - 72 -](#_Toc445823236)

[Figura 3‑15: Promedio en bytes del uso del recurso RAM – prototipo Select - 74 -](#_Toc445823237)

[Figura 3‑16: Promedio del uso Almacenamiento – prototipo Select - 76 -](#_Toc445823238)

[Figura 3‑17: Promedios de eficiencia por cada RDBMS ligero. - 79 -](#_Toc445823239)

[Figura 4‑1: Arquitectura LiriosFadmin - 91 -](#_Toc445823240)

[Figura 4‑2: Arquitectura LiriosDesktop - 92 -](#_Toc445823241)

[Figura 4‑3: Arquitectura LiriosMobile - 93 -](#_Toc445823242)

[Figura 4‑4: Diagrama de componentes LiriosWebService - 94 -](#_Toc445823243)

[Figura 4‑5: Modelo físico de la base de datos LiriosFadminDB - 95 -](#_Toc445823244)

[Figura 4‑6: Inicio de sesión LiriosDesktop - 96 -](#_Toc445823245)

[Figura 4‑7: Detalle de productos y servicio - 96 -](#_Toc445823246)

[Figura 4‑8: Inicio de sesión aplicación móvil - 97 -](#_Toc445823247)

[Figura 4‑9: Diagrama de Burn Down - 99 -](#_Toc445823248)

**ÍNDICE DE ABREVIATURAS**

|  |  |
| --- | --- |
| **ACID** | Atomicidad, Consistencia, Aislamiento, Durabilidad |
| **API** | Interfaz de Programación de Aplicaciones |
| **BD** | Base de Datos |
| **CPU** | Unidad Central de Procesamiento |
| **HTTP** | Protocolo de Transferencia de Hipertexto |
| **HTML** | Lenguaje Hipertexto de Marcas |
| **IDE** | Entorno de Desarrollo Integrado |
| **IIS** | Internet Información Services |
| **JSON** | JavaScript Object Notation |
| **LINQ** | Language Integrated Query |
| **RDBMS** | Sistema de Manejo de Base de Datos Relacional |
| **REST - RESTFUL** | Representational State Transfer |
| **RAM** | Random Access Memory |
| **SGDB** | Sistema de Gestión de Bases de Datos |
| **SQL** | Lenguaje de Consulta Estructurado |
| **SDK** | Software Development Kit |
| **SVC** | Extensión de archivo utilizado por Windows Communication Foundation Microsoft |
| **SOAP** | Simple Object Access Protocol |
| **SO** | Sistema Operativo |
| **SQLSCE** | Sql Server Compact Edition |
| **T-SQL** | Transact SQL |
| **TIC** | Tecnologías de la Información y Comunicación |
| **WP** | Windows Phone |
| **WCF** | Windows Communication Foundation |
| **XML** | Lenguaje de Marcas Extensible |

# RESUMEN

En la actualidad en el hotel “Los Lirios” se presentan necesidades diariamente en el proceso de facturación y gestión de pedidos, problemática que genera la pérdida de tiempo y molestias al cliente al no contar con un sistema informático que automatice estos procesos los cuales se realizan de manera manual dando la posibilidad de que se comentan errores al momento de registrar el detalle por concepto de hospedaje y servicios a la habitación, por tal motivo el hotel desea implementar un sistema de facturación con un módulo de gestión de pedidos como aplicación móvil para prestar un mejor servicio. Esta investigación comprende el análisis comparativo en eficiencia en el consumo de recursos hardware (CPU tiempo, CPU unidades de procesamiento, RAM y Almacenamiento, definidos como parámetros de comparación) entre los RDBMS ligeros SQLite y SQL Server Compact en Windows Phone; el RDBMS ligero que menos consuma estos recursos será el más eficiente y con este se implementa el módulo móvil para la gestión de pedidos del sistema de facturación. El experimento fue diseñado creando los prototipos de pruebas por cada RDBMS ligero con las operaciones Insert, Update, Delete, Select para su medición. Con la ejecución de cada prototipo a través de la herramienta de análisis Profiling Tools del IDE de desarrollo Visual Studio 2013 se recolectó los datos y con el software SPSS se aplicaron métodos estadísticos con el que se procesó la información obtenida con un nivel de significancia del 95 % y un margen de error del 5% de este modo se constató que si existe diferencia significativa entre las muestras de datos ya que SQLite presentó un mejor porcentaje en eficiencia con el 96,09 respecto a SQL Server Compact con el 60,95; Por lo tanto como resultado de la investigación SQLite es más eficiente que SQL Server Compact en el consumo de estos recursos. Por medio de la metodología ágil SCRUM se procedió al desarrollo del sistema de facturación del hotel “Los Lirios” y el modulo móvil para la gestión de pedidos. Se logró con este software resolver las necesidades que presentaba el hotel, mejorando así la calidad de servicio y la satisfacción del cliente.

**Palabras Clave:** <VISUAL STUDIO>, <SQLITE>, <SQL SERVER COMPACT>, <WINDOWS PHONE>, <SCRUM>, <RAM>, <CPU>, <ALMACENAMIENTO>.

# SUMARY

Today, at the hotel "Los Lirios" arise needs daily in the process of billing and order management, the inconvenience generated by the waste of time and discomfort to the client as they do not have a computer system that automates these processes, which are performed manually giving the possibility generating errors when registering the detail by way of accommodation and room services. For this reason, the Hotel want to implement a billing system and a module for order management as an mobile application to provide better service. This research includes a comparative analysis of the efficiency in the use of hardware resources (CPU time, CPU processing units, RAM and storage, defined as parameters for comparison) between light RDBMS SQLite and SQL Server Compact on Windows Phone; the RDBMS that uses less these resources will be the most efficient and the mobile module for order management billing system will be implemented. The experiment was designed to create the prototype testing each light RDBMS operations such as Insert, Update, Delete, Select for measuring the performance of each prototype, through the analysis tool Profiling Tools IDE Visual Studio development 2013 data was collected using SPSS software and statistical methods were used in the information obtained with a significance level of 95% and a margin of error of 5% was processed; thus it was found that if there is significant difference between the samples of the data as presented SQLite percent better efficiency with respect to 96.09 SQL Server Compact with 60.95. Therefore as a result of research SQLite it is more efficient than SQL Server Compact in the consumption of these resources. Through agile methodology SCRUM proceeded to develop the billing system hotel "Los Lirios” and mobile module for order management. It was possible with this software meet the needs presented by the hotel, thus improving the quality of service and customer satisfaction.

Keywords: <VISUAL STUDIO>, <SQLITE>, <SQL SERVER COMPACT>, <WINDOWS PHONE>, <SCRUM>, <RAM>, <CPU>, <DATA STORAGE>.

# INTRODUCCIÓN

Los dispositivos móviles han contribuido para que la sociedad se mantenga comunicada, no solo por llamadas y SMS, sino también mediante el acceso a internet que nos permite estar en línea por medios como el chat o mensajeros, correos electrónicos, video-llamadas y las famosas redes sociales como: Twitter, Facebook, MySpace, entre otros.

Otro aspecto positivo es el acceso móvil a la información ya que ha aumentado su importancia en el entorno de trabajo gracias a la capacidad de acceder y gestionar documentos, y no podemos dejar atrás la comodidad que nos proporcionan debido a que podemos navegar por la red cuando queramos sin importar el espacio en que nos encontremos.

El presente trabajo de investigación para la obtención del título como ingeniero informático, consta en determinar el RDBMS ligero de mayor eficiencia en el consumo de recursos hardware para el desarrollo de un sistema de facturación con un módulo de gestión de pedidos como aplicación móvil para el hotel Los Lirios.

El hotel Los Lirios es una microempresa que se encarga de dar servicio de hospedaje y, alimentación a sus clientes, el avance tecnológico de los dispositivos móviles inteligentes despertó la curiosidad preguntándose cómo podría hacer para que el cliente esté siempre informado de los servicios que ofrece el hotel y a cuales de estos servicios echo uso.

Dado que estos dispositivos pueden ser utilizados en muchas áreas como educación, productividad, salud, geolocalización, etc. y gracias a los componentes con los que vienen integrados, el hotel decidió utilizar los teléfonos inteligentes para mejorar la productividad y la calidad de sus servicios, automatizando el proceso de facturación y gestión de pedidos, en este último es donde intervendrá el aplicativo móvil que se incluirá en el sistema de facturación del hotel.

Se propone esta investigación para comparar los RDBMS ligeros SQLite y SQL Server Compact, para elegir el de mejor eficiencia en el consumo de recursos hardware, para lo cual se establecerán parámetros de comparación, los mismos que serán analizados y estudiados.

En el Capítulo I - Marco Teórico, se detallan diferentes conceptos y términos relacionados con el objeto de estudio de esta investigación.

El Capítulo II - Marco Metodológico, trata sobre los métodos, técnicas, tipo de investigación y pruebas que se realizaron en el desarrollo de esta investigación.

El Capítulo III - Resultados, Discusión y Análisis de Resultados, trata sobre la evaluación de resultados obtenidos permitiendo determinar que RDBMS ligero brinda una mejor eficiencia en el consumo de recursos hardware y así comprobar la hipótesis en base a los resultados obtenidos.

En el Capítulo IV – Sistema de facturación y administración del hotel Los Lirios, se detalla el desarrollo del sistema de facturación, que a su vez contiene el módulo móvil para la gestión de pedidos con el RDBMS ligero de mayor eficiencia en el consumo de recursos hardware.

# Justificación

En el siguiente apartado se definen las razones del tema propuesto, ya sea de manera teórica, metodológica y práctica indicando la importancia de la investigación.

## Justificación teórica*.*

La presente investigación pretende comprobar cuál de los RDBMS ligeros SQLite y SQL Server Compact es más eficiente en el consumo de recursos hardware para su uso en un aplicativo móvil en Windows Phone.

En las empresas de desarrollo por lo general siempre buscan mejorar su desempeño en los proyectos actuales y futuros de desarrollo de software, por ello el problema de escoger una tecnología, servicio o producto aplicable que permita obtener un producto de software ligero, compacto y rápido que ofrezca un abanico de servicios útiles al usuario.

En la actualidad hay varios sistemas operativos que están en constante desarrollo y evolución, IDC empresa de análisis de datos indica que Windows Phone posee un 2,6% de cuota de marcado en la actualidad en lo que va del 2015. La adición de su gran alianza con Nokia detrás de la plataforma ha sido un gran éxito con una significativa parte de las ventas de Windows Phone logrando un 76.0%, además de otros socios que han estado experimentando con Windows Phone y Android al mismo tiempo. Este sistema operativo experimentó un ligero repunte de la cuota hasta el 2,7% interanual desde el 2,5% del 2014 con 9,2 millones de unidades vendidas. Desde la adquisición de Nokia en 2014, Microsoft ha estado renovando la gama de productos con los dispositivos marca Lumia (IDC, 2015).

Se escogió Windows Phone como sistema operativo debido a que tiene un gran crecimiento en diferentes mercados del mundo, capturando poco a poco más adeptos. Hoy en día es el tercer sistema operativo móvil superando a BlackBerry OS que ostentaba el tercer puesto hasta hace pocos meses, además de superar en ventas en algunos países al iPhone de Apple   
(IDC, 2015).

Otra de las razones por la que se escoge este sistema operativo es que está orientado para trabajar con dos motores de bases de datos, los cuales han sido mencionados anteriormente como objeto de estudio.

En lo que respecta a SQLite y SQL Server Compact existen comparaciones basadas en la experiencia de desarrolladores con poco fundamento científico, es decir cada comparación se inclina con el favoritismo hacia un RDBMS, perdiendo objetividad en la comparación, por ende se propone realizar un estudio comparativo de la eficiencia en el consumo de recursos hardware del RDBMS ligero libre SQLite dada su gran distribución para varias plataformas y lenguajes de programación contra el RDBMS ligero propietario SQL Server Compact debido que viene incorporado por defecto en Windows Phone.

## Justificación Metodológica.

A través del método científico, se llevará a cabo el estudio individual de los RDBMS ligeros SQLite y SQL Server Compact Edition en Windows Phone, en donde se procederá con la construcción de 8 prototipos (aplicaciones móviles; 4 SQLite y 4 SQL Server CE) con igual funcionalidad, con cada uno se obtendrán datos importantes que serán analizados y de esta manera comprobar la hipótesis planteada. Se consideró la utilización de una metodología ágil como SCRUM para el desarrollo del sistema de facturación debido a que enfatiza dar pequeños pasos en el desarrollo inspeccionando tanto el producto resultante como la eficacia del producto final.

## Justificación Práctica.

Con la construcción de los prototipos mencionados, permitirán el análisis de la información recolectada entre los RDBMS ligeros SQLite y SQL Server Compact para determinar el de mayor eficiencia en el consumo de recursos hardware y de esta forma el mejor postor formará parte del desarrollo del módulo móvil de gestión de pedidos del sistema de facturación del Hotel Los Lirios.

Para solventar el problema de solicitud de un servicio a la habitación, en la recepción se propone realizar una aplicación móvil que aproveche la infraestructura de red inalámbrica que posee todo el edificio para realizar cualquier pedido desde la comodidad de la habitación, visualizar pedidos, valor a pagar.

Con el sistema de facturación se pretende automatizar el registro y cálculo de valores a pagar por concepto de servicios que solicitan los clientes, gestionar y dar mantenimiento a las habitaciones que son utilizadas y almacenar de manera ordenada la información, mejorando la calidad de servicio al cliente.

Como visión general del sistema se tiene:

* Gestionar los diferentes tipos de usuarios: Cliente, Empleado (Recepcionista, Camarera, Gerente).
* Registrar una habitación para incorporar servicios al hotel.
* Dar de baja a una habitación para tener la información actualizada sobre la disponibilidad de las habitaciones.
* Registro de productos para la venta en el hotel.
* Visualizar la cantidad total generada por concepto de hospedaje para los cobros correctos.
* Gestión de mantenimiento de las habitaciones.
* Actualización de la información de huéspedes.
* Gestionar el stock de productos para llevar un control y actualización del listado de lo que se vende en el hotel.
* Visualización del estado de las habitaciones (disponible, ocupado, mantenimiento, no disponible).
* Registrar los abonos efectuados por el cliente por servicios brindados de hospedaje para su respectiva facturación.
* Visualización de pedidos y valores a pagar por parte del cliente.

La aplicación móvil poseerá las siguientes características:

* Configurar un usuario en la aplicación para la solicitud de pedidos a la habitación.
* Realizar un pedido.
* Visualización de pedidos realizados.
* Visualización del valor a pagar.
* Registro de historial de pedidos del cliente en el dispositivo de manera local.

# Objetivos

Las actividades que se llevarán a cabo en el desarrollo de este trabajo de investigación se dan a conocer a continuación con el objetivo general y los objetivos específicos.

## Objetivo general

Realizar un análisis comparativo entre los RDBMS ligeros SQLite y SQL Server Compact en Windows Phone, para determinar el de mayor eficiencia en el consumo de recursos hardware para el módulo móvil del sistema de facturación del Hotel los Lirios.

## Objetivos específicos

* Estudiar los RDBMS ligeros SQLite y SQL Server Compact en la plataforma Windows Phone.
* Construir prototipos para realizar el análisis comparativo entre los RDBMS ligeros.
* Determinar los parámetros de comparación para establecer el RDBMS ligero de mayor eficiencia en el consumo de recursos hardware.
* Desarrollar el sistema de facturación del Hotel Los Lirios y su módulo móvil con el RDBMS ligero más eficiente.

# Hipótesis

El RDBMS ligero SQLite es más eficiente en el consumo de recursos hardware que SQL Server Compact para el desarrollo del módulo móvil del sistema de facturación del Hotel Los Lirios.

.

# CAPÍTULO I

# MARCO TEÓRICO

En este capítulo se detallan conceptos y definiciones importantes que proporcionan referencias conceptuales utilizadas para el desarrollo de la investigación.

## Windows Phone

Los sistemas operativos para teléfonos móviles se vuelven cada día más importantes, pues la tecnología avanza y en materia de comunicaciones aún más, la telefonía celular cada vez se convierte en una parte importante de nuestras vidas y en una sociedad exigente es importante diseñar sistemas que soporten aplicaciones fluidas, fáciles, accesibles y hasta divertidas.

“Windows Phone es un sistema operativo móvil desarrollado por Microsoft, como sucesor de Windows Mobile, fue diseñado para su uso en teléfonos inteligentes (Smartphone) y está basado en el núcleo del Windows CE 6.0. WP fue reconstruido para ofrecer una mejor experiencia a sus usuarios” (Andrés Rodríguez, 2013).

### Historia

Windows Phone cuyo nombre clave “Photon” era la versión en la que Microsoft trabajaba desde 2004 pero con pocos resultados. Su desarrollo comenzó en el 2008 con la reorganización del equipo de Windows Mobile. Originalmente se calculaba su salida al mercado en el 2009, pero una serie de retrasos obligó a Microsoft a crear una versión de transición de Windows Mobile 6.5 y posponer el lanzamiento de Windows Phone7 (como se conocería a la nueva versión de la plataforma móvil de Microsoft) (Andrés Rodríguez, 2013).

Fue presentado en el Mobile World Congress de Barcelona el 15 de Febrero de 2010 con la finalidad de ser una alternativa real a los sistemas operativos móviles más asentados en el mercado como son iOS y Android. En la actualidad la última versión estable de Windows Phone es la 8.1, pero Microsoft anuncia un nuevo lanzamiento este verano de 2015 y lo da a conocer como Windows 10 tanto para PC’s como para dispositivos móviles unificando la plataforma y dando por terminado el nombre de Windows Phone (Andrés Rodríguez, 2013).

### Versiones de Windows Phone

En el siguiente apartado muestra las diferentes versiones de Windows Phone y sus predecesores:

* **Windows Mobile 6.5**

Primer sistema operativo creado por Microsoft para dispositivos móviles para empresas (PDA, Pocket PC).

* **Windows Phone 7**

Sistema predecesor de Windows Mobile conocida con su nombre clave “Photon” considerada la primera versión de Windows Phone, poseía muchas carencias.

* **Windows Phone 8.0**

Segunda versión de Windows Phone, presentada a finales del 2012, versión que estaba disponible solo para dispositivos nuevos debido a un cambio en el kernel del sistema.

* **Windows Phone 8.1**

Actualización superior de Windows Phone 8, presentada a principios del verano del 2014, con cambios radicales respecto a las versiones anteriores como el centro de notificaciones, asistente personal “Cortana”, sensores de Wifi, datos y batería, incluyendo aplicaciones que vienen en el paquete de instalación como Salud y ejercicios, Comida y bebida, Viajes y Mapas, mejoras en la pantalla de inicio permitiendo agregar o cambiar los fondos de pantalla.

* **Windows Phone 8.1.1**

Ésta es una actualización menor respecto a la anterior, incluía carpetas vivas en la pantalla de inicio, mejoras en el Browser Internet Explorer, Cortana en más idiomas, mejoras en el Action Center (centro de notificaciones) entre otras.

* **Windows Phone 8.1.2**

Actualización que existe hoy en día, incluye el menú de configuraciones dividido en categorías, también posee 3 idiomas nuevos del Suroeste Asiático.

* **Windows 10 Mobile**

Última versión de Windows Phone, liberada en octubre de 2015 en versión preview para que los usuarios la puedan utilizar.

Para concluir, desde que se hizo el lanzamiento de Windows Phone en el 2010 obtuvo cambios notables hasta hoy, paso a paso mejorando la funcionalidad llegando al nivel de los sistemas de mayor renombre en el mercado como son iOS y Android.

### Características principales de Windows Phone

Este apartado trata de las principales características que ofrece este sistema operativo las cuales se mencionan a continuación.

* Soporta varios núcleos y varias resoluciones de pantalla, también ofrece soporte para la tecnología de pagos móviles NFC y gráficos avanzados.
* Posee una interfaz bastante fácil e intuitiva con un sistema de ventanas modulares que ofrecen animaciones y avisos atractivos.
* Posee un servicio de sincronización y la posibilidad de interacción con los archivos office.
* Se basa en el kernel NT de Windows 8, aunque el código ha sido modificado para apoyar a los componentes típicos de un Smartphone. Con la segunda generación de la plataforma, Microsoft ha actualizado los requisitos mínimos de Windows Phone, permitiendo a fabricantes ofrecer más opciones a los consumidores.
* Soporta los procesadores multi-núcleo y cuatro resoluciones de pantalla: 480 × 800, 1280 × 720, 1280 × 768 y 1920 × 1080 píxeles. Los dispositivos deben tener al menos 512 MB, 4 GB de memoria flash, un GPU con soporte DirectX 11, cámara trasera, un módulo Wifi 802.11 b/g/n, Bluetooth, GPS, micro USB, micro SD, acelerómetro, sensor de proximidad y sensor de luz ambiental.
* Geolocalización, mapas, transferencias en segundo plano, edición multimedia, iconos y notificaciones, grabadora de pantalla, cámara, bluetooth, calendarios, compatibilidad con pantallas externas, entre otras características.

### SDK (Software Development Kit - Kit de desarrollo de software).

“Es un conjunto de herramientas y programas de desarrollo que permite al programador crear aplicaciones para un determinado paquete de software, plataforma de hardware, sistema de computadora, consola de videojuegos, sistema operativo o similar” (ALEGSA, 2010).

Microsoft provee de manera gratuita el SDK con todo lo necesario para desarrollar aplicaciones para la plataforma de Windows Phone desde la página oficial permitiendo así su descarga, pero para ello se necesita cumplir con algunos requisitos como tener instalado el IDE de desarrollo Visual Studio 2013 o superior bajo licencia, este incluye nuevos emuladores para los distintos tamaños de pantalla de los dispositivos en el mercado y plantillas para las aplicaciones universales.

A continuación se presenta las herramientas que se utilizan para el desarrollo de aplicaciones para Windows Phone:

* SDK de Windows 8.1 o superior
* Visual Studio 2013 o superior.

Para concluir, las herramientas del SDK de Windows Phone dan un gran aporte a la creación de aplicaciones mejorando la eficacia del desarrollador. Existe un gran soporte en documentación y ejemplos para introducirse rápidamente al desarrollo en Windows Phone. Uno de los aspectos negativos del SDK de Windows Phone es no ser multiplataforma solo siendo posible su instalación en Windows.

### Lenguajes de programación

Con el SDK de Windows Phone se logra la creación de aplicaciones que soportan los lenguajes de programación C# o Visual Basic haciendo uso de XAML para interfaces gráficas, C++ con Directx para el desarrollo de video-juegos, JavaScript con HTML para el desarrollo de interfaces gráficas. (Ricardo J, 2012).

### Silverlight para Windows Phone

Es una de las plataformas para el desarrollo de aplicaciones en Windows Phone, aplicaciones basadas en XAML que contengan transiciones y efectos visuales. Silverlight es una tecnología multi-navegador de .NET Framework para la construcción de experiencias multimedia y aplicaciones interactivas para la web en equipos de escritorio y también disponible para Windows Phone, incluye un tiempo de ejecución optimizado para visualizar contenido en dispositivos con memoria limitada, se encuentra alojado en el dispositivo cliente (Silverlight for Windows Phone, Microsoft 2015).

### Windows Phone - Universal Apps

“Es un gran avance tecnológico desarrollado por Microsoft, permitiendo que las plataformas de Windows Phone y Windows converjan, es decir que comparten el mismo código en el desarrollo de una aplicación” (Gallardo Sales Sergio, 2014).

Experiencia de Windows que está integrada en la Windows Universal Plataform (UWP, Plataforma Universal de Windows), que se presentó en Windows 8 por primera vez como Windows en tiempo de ejecución. Las aplicaciones universales se distribuyen la mayoría de veces a través de la Tienda de Windows. La plantilla de aplicaciones universales de Windows, permite crear fácilmente aplicaciones para la Tienda de Windows y aplicaciones para la Tienda de Windows Phone a partir de un solo proyecto. Todo esto se logra gracias a las versiones actuales de Visual Studio que incluye esta característica, pudiendo así generar los paquetes necesarios mediante una única acción y las aplicaciones llegarán a los usuarios de cualquier dispositivo de Windows. En la Figura 1-1 se muestra el esquema de Universal Apps.

****

Figura 1‑1: Universal apps Windows Platform

**Fuente Web:** https://msdn.microsoft.com/es-es/library/windows/apps/dn894631.aspx

## Sistemas de Gestión de Bases de Datos Relacionales

“Un sistema de bases de datos relacionales RDBMS (Relational Database Management System, por sus siglas en inglés) tales como Oracle, MySQL, SQL Server, PostgreSQL, Informix, entre otros, son aquellos que almacenan y administran de manera lógica los datos en forma de tablas. Una tabla es a su vez, un método para presentar los datos en la forma de filas y columnas, permitiendo ejecutar las tareas que se mencionan a continuación, de una forma entendible y razonablemente” (Diego Pino, 2009):

* Ingresar datos al sistema.
* Almacenar los datos.
* Recuperar los datos y trabajar con ellos.
* Proveen herramientas para capturar, editar y manipular datos.
* Aplicar seguridad.
* Crear reportes e informes con los datos.

### Objetivos de los RDBMS

Entre los objetivos que debe cumplir un RDBMS se mencionan los siguientes (Andrade Mario & Parra Jhony, 2014):

* Integridad de los datos
* Seguridad de la información
* Independencia de los datos
* Reducir al mínimo la redundancia de los datos
* Permitir el control de concurrencia y simultaneidad.

### RDBMS tipo compacto

“Permiten almacenar información en dispositivos de una forma sencilla, eficaz, potente, rápida y en equipos con pocas capacidades de hardware, como puede ser una PDA o un teléfono celular. SQLite y SQL Server CE forman parte de este grupo de RDBMS, agregan extensiones que facilitan su uso en cualquier ambiente de desarrollo. Esto permite que estos motores de bases de datos soporten desde las consultas más básicas hasta las más complejas del lenguaje SQL, y lo más importante es que se pueden usar tanto en dispositivos móviles como en sistemas de escritorio sin necesidad de realizar procesos complejos de importación y exportación de datos, ya que existe compatibilidad al 100%, haciendo que la portabilidad entre dispositivos y plataformas sea transparente “(Filein Romel, 2007).

## SQL Server Compact Edition para Windows Phone

“Generalmente los requerimientos de las bases de las bases de datos para almacenar información nos empujan a utilizar motores donde el rendimiento, la integridad referencial, la concurrencia, son factores determinantes en la selección de un RDBMS. Sin embargo, hay ocasiones en donde los requerimientos son otros, como la portabilidad, escasa memoria, rapidez, entre otros, es allí donde han surgido pequeños motores con las características de algunos grandes. Este tipo de motores son denominados bases de datos incrustadas (empotradas, embebidas o compactas), que se alinean a los requerimientos antes mencionados” (José Gabriel Osorio, 2010).

SQL Server Compact es una base de datos compacta que se puede implementar en equipos de escritorio, dispositivos inteligentes, Tablet PC y equipos portátiles proporcionando a los desarrolladores un modelo de programación común a otras ediciones de SQL Server para el desarrollo tanto de aplicaciones nativas como administradas. Ofrece funcionalidad de base de datos relacional en un espacio reducido, un sólido almacén de datos, un procesador de consultas y una conectividad confiable (Lo nuevo de SQL Server Compact Edition, 2015).

### Historia

En el año 2001 Microsoft lo dio a conocer como SQL Server CE 1.0, un servicio de almacenamiento relacional que permitía la sincronización de aplicaciones móviles para dispositivos como el Pocket PC, estas desarrolladas con Visual Studio y C# Embedded con SQL Server 2000. Cuando Microsoft lazó Visual Studio .NET 2003 y .NET Compact Framework 1.1, presentó de igual forma su nueva versión compacta de SQL Server CE 2.0. En el mes de Noviembre del 2006 dio el lanzamiento oficial de Visual Studio 2005 con SQL Server 2005, donde incluía la versión 3.0 con el nombre de SQL Server Mobile 3.0 con características y funcionalidades nuevas tales como acceso simultáneo y soporte transaccional, todo esto integrado con Visual Studio y su herramienta de gestión SQL Management Studio.

Con el nombre de SQL Server Everywhere (nombre en clave) se hace un su nuevo lanzamiento, versión con soporte multiplataforma para sistemas operativos basados en Windows 32/64 bits, oficialmente llamada SQL Server Compact 3.1.

SQL Server Compact 3.5 SP1 nació junto a Visual Studio 2008.Net, luego denominado SQL Server Compact 3.5 SP2 con Visual Studio 2010 con el nuevo lanzamiento de este IDE y por último se conoce a SQL Server Compact Edition 4.0 con el lanzamiento de Visual Studio 2012 y SQL Server 2012.

### Características

SQL Server CE es una base de datos idónea para aplicaciones de dispositivos móviles ya que proporciona:

* Operación ligera (se ejecuta en proceso, desde la versión de Windows Phone 7 ya viene incluido con el SO)
* Posee un subconjunto enriquecido de características de programación de consulta de SQL Server (hermano mayor).
* Posee un subconjunto enriquecido y coherente de tipos de datos de SQL Server.
* Almacenamiento local de datos mediante un archivo de bases de datos con extensión \*.sdf.
* Seguridad local.

### Lenguaje de consulta

El lenguaje de consulta utilizado por SQL Server CE es LINQ to SQL (Language-Integrated Query por sus siglas en ingles), que es un componente de .NET Framework 3.5 que proporciona una infraestructura en tiempo de ejecución para administrar datos relacionales como objetos, sin perder la capacidad de realizar consultas. Para ello, LINQ to SQL traduce a SQL (lenguaje de consulta estructurado) las consultas integradas en el lenguaje para su ejecución por parte de la base de datos y posteriormente vuelve a trasladar los resultados tabulares a los objetos que se definan (LINQ to SQL Server Compact, 2015).

LINQ to SQL es utilizado para acceder a los datos de la BD local y no a través de Transact-SQL ya que no está permitido por problemas de compatibilidad.

### Arquitectura de SQL Server CE

El siguiente apartado trata sobre la arquitectura de SQL Server CE en Windows Phone (Bases de datos locales para Windows Phone - Panorama arquitectónico, 2015):

Para la creación de aplicaciones en Windows Phone y la utilización de una Base de Datos embebida, compacta o ligera, para almacenar y recuperar datos en una base de datos local, se utiliza de igual forma LINQ to SQL como se hace mención en el párrafo anterior. Este proporciona un enfoque orientado a objetos para trabajar con datos y consta de un modelo de objetos y un tiempo de ejecución. El modelo de objetos de LINQ to SQL se compone principalmente por el objeto System.Data.Linq.DataContext (que es el contexto de datos), que actúa como un proxy para la base de datos local.

Este contexto de datos (DataContext) se lo debe crear para poder tener acceso a la base de datos local en Windows Phone, aquí es necesario aclarar, que no se puede utilizar ADO.NET de manera directa para el acceso a la base de datos local. Por este motivo se debe crear un contexto de datos de la BD, este DataContext, no es más que una clase en uno de los lenguajes de la plataforma .NET (VB.NET o C#), en la cual se encuentran métodos que permiten las operaciones CRUD (Create, Read, Update, Delete), clases de tipo entidad que representan un registro de cada una de las tablas, y las clases de tipo colección que representan la tabla o un conjunto de registros de la tabla.

LINQ to SQL, es el mecanismo utilizado por SQL Server CE para todas las operaciones con la base de datos que es almacenada de modo aislado dentro de la aplicación (isolated storage). Entre las operaciones se incluyen tanto las de consulta y modificación de datos así como las de definición o creación del propio esquema de la base de datos. Todo lo mencionado anteriormente da origen al modelo de arquitectura que se representa en la figura 1-2:



Figura 1‑2: Arquitectura SQL Server CE

**Fuente Web:** <https://msdn.microsoft.com/en-us/library/windows/apps/hh202860(v=vs.105).aspx#BKMK_UsingtheDatabase>

### Ventajas y Limitaciones de SQL Server Compact Edition

Existen escenarios en donde contar con una base de datos relacional es justamente uno de los requerimientos básicos, por ejemplo: el tener catálogos de datos locales que puedan ser utilizados cuando la señal de datos del teléfono se pierda, o un caché local de datos que posteriormente puedan ser sincronizados a un almacenamiento más robusto y persistente (en la nube, por ejemplo), SQL Server CE entra en este campo en donde citaremos algunos de sus beneficios y limitaciones:

Ventajas (Ventajas y limitaciones de SQL Server Compact, 2015):

* Es un completo y versátil RDBMS ligero para Windows Phone en términos de potencialidad y facilidad de uso.
* Permite la validación y posee una gran cantidad de tipos de datos similar a su hermano mayor Microsoft SQL Server.
* SQL Server CE no corre como un servicio en segundo plano dentro del teléfono, sino que únicamente en el mismo proceso de la aplicación que lo invoca.

Limitaciones (Ventajas y limitaciones de SQL Server Compact, 2015):

* No permite Transact-SQL directo.
* No posee un diseñador visual.
* Posee un rendimiento no muy aceptable en presencia de grandes cantidades de datos.
* Es compatible solo con las plataformas Microsoft Windows (Desktop) y Windows Phone, no es portable sobre otras plataformas.

### Tipos de datos soportados por SQL Server CE

SQL Server CE admite los siguientes tipos de datos que se muestran en la tabla 1-1:

Tabla 1‑1: Tipos de datos para SQL Server CE.

|  |  |
| --- | --- |
| Tipo de datos | Descripción |
| bigint | Datos enteros (número entero) de –2^63 (–9.223.372.036.854.775.808) a 2^63–1 (9.223.372.036.854.775.807). El tamaño de almacenamiento es de 8 bytes. |
| integer | Datos enteros (número entero) de –2^31 (–2.147.483.648) a 2^31–1 (2.147.483.647). El tamaño de almacenamiento es de 4 bytes. |
| smallint | Datos enteros de –32.768 a 32.767. El tamaño de almacenamiento es de 2 bytes. |
| tinyint | Datos enteros de 0 a 255. El tamaño de almacenamiento es de 1 byte. |
| bit | Datos enteros con el valor 0 ó 1.  El tamaño de almacenamiento es de 1 bit. |
| numeric (p, s)  Synonyms:  decimal(p,s) and dec (p,s) | Datos numéricos de escala y precisión fija de –10^38+1 a 10^38–1. La variable *p*especifica la precisión y puede variar entre 1 y 38. La variable *S* especifica la escala y puede variar entre 0 y *p*.  El tamaño de almacenamiento es de 19 bytes. |
| money | Valores de datos de moneda de (–2^63/10000) (–922.337.203.685.477,5808) a 2^63–1 (922.337.203.685.477,5807), con una precisión de una diezmilésima de la unidad monetaria. El tamaño de almacenamiento es de 8 bytes. |
| float | Datos numéricos de punto flotante de –1,79E +308 a 1,79E+308  El tamaño de almacenamiento es de 8 bytes. |
| real | Datos numéricos de precisión flotante de –3,40E+38 a 3,40E+38.  El tamaño de almacenamiento es de 4 bytes. |
| datetime | Datos de fecha y hora desde el 1 de enero de 1753 hasta el 31 de diciembre de 9999, con una precisión de una tricentésima de segundo o 3,33 milisegundos. Los valores se redondean en incrementos de 0,000, 0,003 ó 0,007 milisegundos. |
| national character(n)  Synonym:nchar(n) | Datos Unicode de longitud fija con una longitud máxima de 4000 caracteres. La longitud predeterminada es 1. El tamaño de almacenamiento, en bytes, es el doble del número de caracteres especificado. |
| national character varying(n)  Synonym:nvarchar(n) | Datos Unicode de longitud variable con una longitud entre 1 y 4000 caracteres. La longitud predeterminada es 1. El tamaño de almacenamiento, en bytes, es el doble del número de caracteres especificado. |
| ntext | Datos Unicode de longitud variable con una longitud máxima de (2^30–2)/2 (536.870.911) caracteres. El tamaño de almacenamiento, en bytes, es el doble del número de caracteres especificado.Nota |
| nchar | Datos Unicode de longitud fija de n caracteres. n debe ser un valor comprendido entre 1 y 4.000. El tamaño de almacenamiento es el doble de n bytes. |
| binary(n) | Datos binarios de longitud fija con una longitud máxima de 8000 bytes. La longitud predeterminada es 1.  El tamaño de almacenamiento es fijo y corresponde a la longitud en bytes declarada en el tipo. |
| varbinary(n) | Datos binarios de longitud variable con una longitud máxima de 8000 bytes. La longitud predeterminada es 1. |
| image | Datos binarios de longitud variable con una longitud máxima de 2^30–1 (1.073.741.823) bytes.  El tamaño de almacenamiento es la longitud del valor en bytes. |
| uniqueidentifier | Identificador único global (GUID). El tamaño de almacenamiento es de 16 bytes. |
| IDENTITY [(s, i)] | Es una propiedad de una columna de datos, no un tipo de datos distintivo.  Para las columnas de identidad, solo se pueden utilizar las columnas de datos de los tipos de datos enteros. s (inicialización) = valor inicial  i(incremento) = valor de incremento. |

**Realizado por:** Darwin Cedeño y Diego Quinatoa 2015

**Fuente**: Tipos de datos SQL Server Compact, 2015

### Instalación de SQL Server CE.

SQL Server CE viene incluido en modo nativo en el sistema operativo solo para aplicaciones que utilizan el API Windows Phone Silverlight desde las versiones de Windows Phone 7 .x hasta la versión 8.1; en aplicaciones que utilizan la API de Windows Runtime (Universal Apps) no existe el API de SQL Server CE. Esto quiere decir que no existe un proceso setup directo para la instalación de este motor de bases en el dispositivo.

## SQLite para Windows Phone

El uso de las bases de datos ya se ha extendido de los servidores hacia los dispositivos móviles. El desarrollo constante de la tecnología conjuntamente con los nuevos requerimientos de las empresas ha llevado a crear diversos métodos de almacenamiento de información en dispositivos móviles. SQLite es una biblioteca que implementa un motor de base de datos SQL independiente, sin servidor, configuración cero, transaccional. El código es de dominio público y por lo tanto es libre de uso para cualquier propósito comercial o privado. Es uno de los RDBMS más desplegados en el mundo de las base de datos con una variedad de aplicaciones que hacen uso de ella, incluyendo varios proyectos de alto perfil (Acerca de SQLite, s.f.).

En la actualidad, la demanda de bases de datos para dispositivos móviles como PDAs y teléfonos celulares ha crecido exponencialmente en los últimos años debido a las necesidades de las empresas de tener la información al instante. Esta necesidad ha provocado que el almacenamiento de los datos en estos dispositivos haya mejorado tanto en capacidad como en herramientas. Gracias a esto, actualmente se cuenta con diversas opciones de manejadores de bases de datos para móviles, y una de estas es SQLite, otro de los objetos de estudios de esta investigación.

Es una biblioteca compacta con todas las funciones habilitadas, el tamaño de la biblioteca puede ser inferior a 500KB, dependiendo de la configuración de optimización de la plataforma y del compilador de destino. Si se omiten características opcionales, el tamaño de la biblioteca de SQLite puede reducirse debajo de 300KB (Acerca de SQLite, s.f.).

SQLite es una opción de motor de bases de datos popular en dispositivos de memoria limitada como teléfonos móviles, hablamos en que el rendimiento suele ser bastante bueno incluso en entornos de baja memoria.

### Historia

De acuerdo a lo que cuenta Dwayne Richard Hipp (creador de SQLite), cuando pertenecía a la fuerza naval de los Estados Unidos como desarrollador de software expresa lo siguiente:

*“El proyecto SQLite surgió de una necesidad personal, para mi propio uso. En enero de 2000 D. Richard Hipp estaba trabajando con su equipo en la Fuerza naval de los Estados Unidos, en un proyecto de software, el cual se conectaba a una base de datos Informix, el motor funcionaba muy bien, pero habían tenido problemas para hacer una reconfiguración cuando el sistema se reiniciaba. Luego cambiaron a Postgre SQL, pero administrar la base de datos era un problema. Si bien aún estaba luchando para hacer frente a estas cuestiones, surgió la idea de escribir un simple motor de base de datos SQL que permitiera leer los archivos del disco duro, y luego ser llamados en diferentes solicitudes. Cinco meses más tarde, se quedó sin un contrato por algunos meses y por lo que comenzó a escribir las primeras versiones de lo que hoy se conoce como SQLite, con el pensamiento de que sería útil en algún problema similar a los que el tubo”* (ANTONIO CABERO, s.f. p2.)

### Características de SQLite

Las características más relevantes de este RDBMS ligero son (Características de SQLite, 2015):

* En SQLite [las](http://sqlite.org/transactional.html) transacciones  son atómicas, coherentes, aislado y duradero (ACID), incluso después de los fallos del sistema y fallas de energía.
* Es una base de datos completa que se almacena en un único archivo.
* Puede funcionar enteramente en memoria lo que la hace muy rápida
* La implementación es completa, con características avanzadas como índices parciales y expresiones de tabla comunes.
* Disponible como un solo archivo de código Fuente que es fácil de recopilar y por lo tanto es fácil de añadir en un proyecto más amplio.
* [Autónomo](http://sqlite.org/selfcontained.html) : sin dependencias externas.
* Multiplataforma: Android, BSD, iOS, Linux, Mac, Solaris, VxWorks, y Windows (Win32)
* Las Fuentes de SQLite son de [dominio público](http://sqlite.org/copyright.html) permitiendo su utilización para cualquier propósito.
* Viene con una interfaz independiente  [de líneas de comandos](http://sqlite.org/cli.html) Shell (CLI - siglas en ingles) en donde el cliente puede utilizar para administrar las bases de datos SQLite.
* Soporta texto en formato UTF-8 y UTF-16, así como datos numéricos de 64 bits.

### Lenguaje de consulta para SQLite

SQLite entiende la mayor parte del lenguaje SQL (Lenguaje de Consulta Estructurado) estándar y al mismo tiempo añade características propias (*Sentencias y lenguaje de consulta SQL en SQLite*, SQLite org, 2015).

En la 1-3 se muestra una lista de sentencias SQL aceptadas por SQLite:

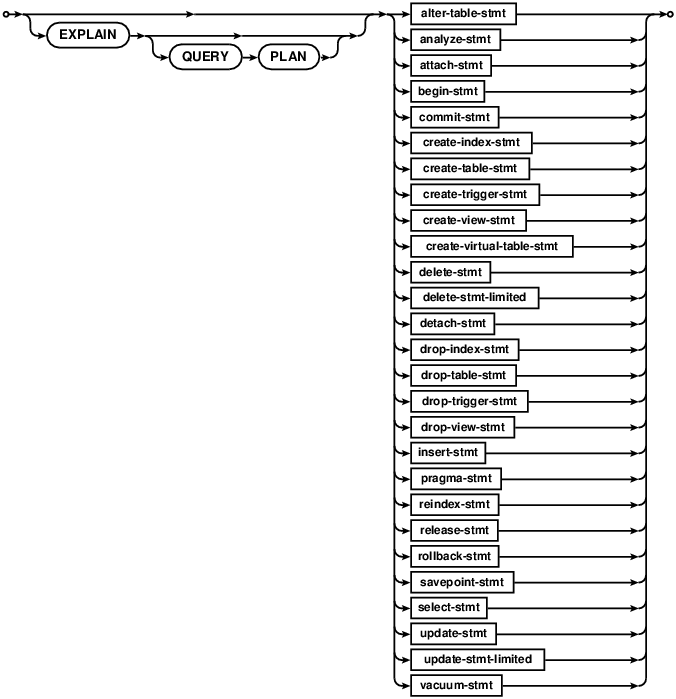


Figura 1‑3: Sentencias SQL

Fuente Web: <https://www.sqlite.org/lang.html>

### Arquitectura de SQLite

El diagrama de bloques que se ilustra en la figura 1-4 muestra los componentes principales de SQLite y cómo se interrelacionan. El texto que sigue proporcionará una visión general de cada uno de estos componentes (Arquitectura de SQLite, 2015):

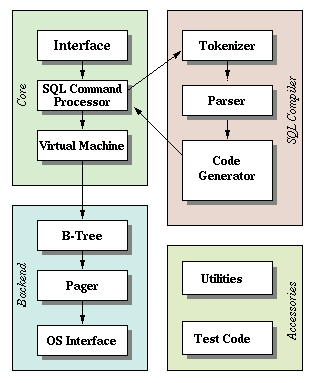
****

Figura 1‑4: Diagrama de bloques – Arquitectura SQLite

Fuente Web: [http://sqlite.org/arch.html](http://sqlite.org/arch.html%20)

**Interface:** Gran parte de la interfaz pública de SQLite se implementa mediante funciones que se encuentran en el main.c, legacy.c y vdbeapi.c; archivos de código Fuente, aunque algunas rutinas están esparcidos en otros archivos en el que se puedan tener acceso a las estructuras de datos con este ámbito.

**Tokenizer**: Cuando una cadena que contiene instrucciones SQL son ejecutados, la interfaz pasa la cadena al tokenizer. El trabajo del tokenizer es romper la cadena en tokens y pasar los tokens uno por uno al analizador.

**Parser (analizador):** Es la pieza que asigna significado a los símbolos basados en un contexto. El analizador de SQLite se genera mediante Lemon LALR (generador de analizadores sintácticos para C o C ++) que utiliza una sintaxis de entrada que no es propenso a errores.

**Code Generator:** Esta propiedad después de que el analizador(parser) reúne fichas en sentencias SQL completos, llama al code generator para producir código haciendo como trabajo la gestión de las solicitudes de sentencias SQL.

**Virtual Machine:** Implementa un motor de computación abstracta diseñada específicamente para manipular archivos de base de datos. El programa generado por el generador de código es ejecutado por la máquina virtual.

**B-Tree:** Una base de datos SQLite se mantiene en el disco usando una aplicación B-Tree que se encuentra en el archivo de cabecera btree.c, Un árbol B se usa por separado para cada tabla e índice en la base de datos. Todos los árboles B se almacenan en el mismo archivo de disco.

**Page Cache (pager):** El módulo B-tree solicita información del disco en trozos de tamaño fijo. El tamaño del fragmento predeterminado es 1024 bytes, pero puede variar entre 512 y 65536 bytes. El Page Cache se encarga de leer, escribir y almacenar en caché estos fragmentos ofreciendo la reversión y la abstracción atómica de los mismos y a su vez se encarga de bloqueo del archivo de base de datos.

**OS Interface:** Con el fin de proporcionar la portabilidad entre sistemas operativos SQLite utiliza una capa de abstracción para interactuar con el sistema operativo, en donde existen instrumentos específicos de operación que poseen su propia cabecera dependiendo del SO en que se implemente SQLite por ejemplo: (os\_unix.h para Linux, os\_win.h para Windows).

**Utilities:** Se da la asignación de memoria y las rutinas de comparación de cadena que están ubicados en la cabecera útil.c.

**Test Code:** Si se cuenta con un scripts de prueba, más de la mitad del código total base de SQLite se dedica a las pruebas. La interface backend os\_test.c se utiliza para simular fallas de energía para verificar el mecanismo de recuperación de fallos en el localizador de errores.

### Ventajas y Limitaciones de SQLite

Las ventajas y limitaciones que presenta este RDBMS a continuación (Ventajas y limitaciones de SQLite, 2015):

Ventajas:

* SQLite no necesita ser instalado antes de ser utilizado, esto quiere decir que no existe un proceso “Setup”.
* No necesita de un proceso servidor que deba ser iniciado, detenido o configurado.
* No hay necesidad de un administrador para crear una nueva instancia de bases de datos o asignar permisos de acceso a los usuarios.
* SQLite se implementa como un proceso de servidor independiente.
* Es de código abierto.
* Una base de datos SQLite se genera en un solo archivo de disco que puede estar ubicado en cualquier lugar de la jerarquía de directorios.
* El formato de archivo de SQLite es multiplataforma.
* Es de dominio público.
* Portabilidad: SQLite se ejecuta en varias plataformas y puede ser fácilmente portable sin necesidad de alguna configuración en especial.

Limitaciones:

* Limitaciones Where: esta limitación está dada por el soporte para clausulas anidadas.
* No posee clave foránea: se hace caso omiso a las claves foráneas.

### Tipos de Datos soportados

SQLite utiliza un sistema de tipo dinámico general, es decir el tipo de datos de un valor se asocia con el valor en sí, no con su contenedor. El sistema de tipo dinámico de SQLite es compatible con los sistemas de tipo estáticos más comunes al de otros motores de bases de datos, en el sentido de que las declaraciones SQL que trabajan en las bases de datos de tipos estáticos debería funcionar de la misma manera en SQLite. Cada valor almacenado en una base de datos SQLite tiene los siguientes tipos de datos para el almacenamiento como se muestra en la tabla 1-2 (Tipos de datos SQLite, 2015):

Tabla 1‑2: Tipos de datos SQLite.

|  |  |
| --- | --- |
| TIPO DE DATO | DESCRIPCION |
| NULL | Es un valor NULL. |
| INTEGER | El valor es un entero con signo, almacena en 1, 2, 3, 4, 6, o 8 bytes dependiendo de la magnitud del valor. |
| REAL | Es un valor de punto flotante, almacenado como un número de 8 bytes. |
| TEXT | El valor es una cadena de texto, almacenados utilizando la codificación de la base de datos (UTF-8, UTF-16BE o UTF-16LE). |
| BLOB | El valor es una masa de datos, almacenados exactamente como de entrada. |
| BOOLEAN | SQLite no tiene un tipo de dato Boolean separado. En cambio, los valores booleanos se almacenan como enteros 0 (falso) y 1 (verdadero). |
| FECHA Y TIEMPO | SQLite no tiene un tipo de datos específico reservado para el almacenamiento de fechas y / o tiempos, pero se  podría decir que los incorporados en fecha y hora de SQLite son capaces de almacenar estos valores como texto, REAL, o INTEGER |

**Realizado por:** Darwin Cedeño y Diego Quinatoa 2015

**FUENTE**: SQLite, 2015 (Tipos de datos SQLite, 2015).

### Instalación

Por lo general no existe un proceso “Setup” directo que permita la instalación de SQLite en Windows Phone; en este apartado se explica cómo se debe realizar la instalación de la librería y referencias necesarias para el correcto funcionamiento de SQLite en este sistema operativo.

Lo primero que se debe hacer es descargar el SDK para Windows Phone Silverlight 8.0/8.1 v3.8.5 desde el sitio SQLite.org (<http://sqlite.org/2014/sqlite-wp80-winrt-3080500.vsix>) e instalarlo. En Visual Studio abrir la consola de Administrador de paquetes Nuget para proceder con la instalación del paquete SQLitePCL y dentro de la consola digitar la siguiente línea (Install-Package SQLitePCL), luego seleccionar la arquitectura de compilación de destino en la ventana de configuración de la solución de Visual Studio, esto hará referencia a la dll del SDK apropiada al proyecto que se compila (Instalación de SQLite en Windows Phone – Microsoft Open Technologies, 2015).

## Metodologías Ágiles

El desarrollo de software debe cumplir con la característica principal de trabajar con variables o requerimientos desconocidos, permitiendo así crear velozmente prototipos y versiones previas a la entrega final de un producto, otorgándole al cliente una satisfacción para de esta manera tener menor riesgo ante la posibilidad de cambios en los requisitos funcionales.

Considerando la aparición de metodologías denominadas ágiles, las que se adaptan al proceso de desarrollo de software específico y que a la vez permiten fácilmente reunir y contemplar cambios en estos requisitos, brindan una visión clara de lo que se está trabajando dentro del equipo, consiguiendo la participación del cliente en cada momento para lograr los resultados que este desee.

En febrero de 2001 en Utah-EEUU, nace el término "ágil" aplicado al desarrollo de software, en donde participo un grupo de 17 expertos de la industria del software, entre creadores o impulsores de metodologías de software. El objetico era mejorar los valores y principios que deberían permitir a los equipos desarrollar software rápidamente y hacer frente a los cambios se susciten a largo del proyecto pretendiendo así dar una alternativa a los procesos de desarrollo de software tradicionales, caracterizados por ser rígidos y enfocados por la documentación que se genera en cada una de las actividades desarrolladas (Letelier, Penadés, s.f.).

A continuación se presenta una lista de las principales metodologías ágiles:

* SCRUM
* Crystal Methodologies
* Dynamic Systems Development Method
* Adaptive Software Development
* Feature-Driven Development
* Lean Development

### Metodología SCRUM

SCRUM es un marco de trabajo en el que equipos cross-funcionales pueden crear productos o desarrollar proyectos de una forma iterativa e incremental. El desarrollo se estructura en ciclos de trabajo llamados Sprints (también conocidos como iteraciones). Estas iteraciones no deben durar más de cuatro semanas cada una (siendo dos semanas la duración más habitual) y tienen lugar una tras otra sin pausa entre ellas (Dreemer, Benefield, Larman, Vodde, 2012).

El ciclo de vida que define SCRUM es incremental y se caracteriza por ser adaptable, a continuación alguna de las principales características de SCRUM (Andrea Peralta, p2, 2003):

* Equipos auto dirigidos.
* Utiliza reglas para crear un entorno ágil de administración de proyectos.
* No prescribe prácticas específicas de ingeniería.
* Los requerimientos se capturan como ítems de la lista del Product Backlog.
* El producto se construye en una serie de Sprints de un mes de duración.

**Elementos de la metodología SCRUM**

**Product Backlog:** Es una lista priorizada que define el proyecto que se va a realizar, con visión única y definitiva de todo lo que podría ser realizado en algún momento por el equipo de trabajo en orden de prioridad. Sólo existe un product backlog para cada producto, lo cual significa que el dueño del producto debe tomar decisiones de prioridad sobre todo el abanico de opciones, en representación de los intereses de todos los que conforman el team SCRUM. No contiene historias de usuario si no simplemente elementos, estos elementos se pueden expresar como historias de usuario, casos de uso o cualquier otra aproximación a la toma de requisitos que el grupo considere útil (Dreemer, Benefield, Larman, Vodde, 2012).

**Sprints:**

Procedimiento de adaptación de las variables del entorno (requerimientos, tiempo, recursos, conocimiento, tecnología). Son ciclos iterativos en los cuales se desarrolla o mejora una funcionalidad para producir nuevos incrementos. Durante un Sprint el producto es diseñado, codificado y probado y su arquitectura y diseño evolucionan durante el desarrollo (Andrea Peralta, p2, 2003).

**Roles SCRUM**

Existen tres roles para ésta metodología: Dueño del producto, Equipo, Scrum Master, ellos conforman el team SCRUM los que se describen a continuación (Dreemer, Benefield, Larman, Vodde, 2012):

**Dueño del Producto (Product Owner):** Es responsable de maximizar el retorno de inversión (ROI) a base de identificar las funcionalidades del producto, trasladarlas a una lista priorizada, decidir cuáles deberían estar al principio de la lista para el siguiente Sprint, y re priorizar y refinar continuamente dicha lista.

**Equipo**: También llamado equipo de desarrollo, construye lo que el dueño del producto indica; por ejemplo una aplicación o un sitio web. Engloba toda la experiencia y conocimiento necesarios para desarrollar un producto potencialmente entregable en cada Sprint y es “auto-organizado” con un amplio margen de autonomía y responsabilidad.

**Scrum Master:** Ayuda al área de producto a aprender y aplicar Scrum para obtener valor de negocio. Hace todo lo que esté en su mano para ayudar al equipo, al dueño de producto y a la organización a tener éxito. En la Figura 1‑5 se indica el proceso de desarrollo SCRUM

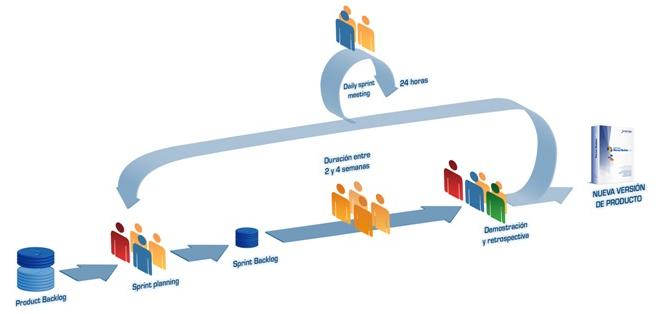


Figura 1‑5: Procesos de la metodología SCRUM

**Fuente Web:** <https://www.softeng.es/es-es/empresa/metodologias-de-trabajo/metodologia-scrum/proceso-roles-de-scrum.html>

El desarrollo se realiza de forma incremental, cada iteración se la denomina Sprint**,** con una duración preestablecida de entre 2 y 4 semanas, obteniendo como resultado una versión de software con prestaciones nuevas listas para ser usadas. En cada nuevo Sprint**,** se va mejorando la funcionalidad ya desarrollada añadiendo nuevas prestaciones dando prioridad a aquellas que aporten mayor valor de negocio (Procesos y roles SCRUM, s.f.).

# CAPÍTULO II

# MARCO METODOLÓGICO



El siguiente capítulo abarca el proceso de obtención los datos, como fueron procesados durante el desarrollo de este trabajo investigativo, para posteriormente realizar el análisis respectivo y determinar el RDBMS ligero de mayor eficiencia.

Uno de los aspectos importantes que se considera en la creación de una aplicación software es la utilización de los recursos hardware que emplea para funcionar de manera óptima.

En lo que respecta a los RDBMS ligeros todos tiene una forma diferente de funcionar, por lo que en el presente trabajo investigativo se realiza un análisis comparativo de la eficiencia en el consumo de recursos hardware entre SQLite y SQL Server CE con el objetivo de conocer el RDBMS ligero que mejor prestaciones ofrece para el desarrollo del módulo de gestión de pedidos como aplicación móvil del sistema de facturación del hotel Los Lirios.

## Tipo de Investigación

Para esta investigación se utilizó el método científico con carácter descriptivo e inferencial, dirigido a determinar el estado de la información recolectada dentro de una población, y a su vez a la toma de decisiones en cuanto a la aprobación o rechazo de la hipótesis que se planteó anteriormente.

## Parámetros de comparación

Para evaluar y comparar los RDBMS SQLite y SQL Server Compact, y elegir el mejor para la implementación del módulo de gestión de pedidos como aplicación móvil “Lirios Mobile” del sistema de facturación se determinaron parámetros a ser analizados tales como: CPU unidades de tiempo, CPU Unidades de procesamiento, RAM, Almacenamiento. Como el objeto de estudio en esta investigación es determinar la eficiencia en el consumo de recursos hardware entre estos RDBMS, se la define como:

“La capacidad de utilizar un recurso de manera racional para llegar a cumplir un objetivo en el menor tiempo posible y con el mínimo uso de estos recursos empleados” (ALEGSA, 2010).  
Analizando lo mencionado se determinó cuatro parámetros de comparación, definidos en la Tabla 2‑1:

Tabla 2‑1: Parámetros de comparación.

|  |  |
| --- | --- |
| PARÁMETROS | DESCRIPCIÓN |
| CPU unidades de tiempo | Proporciona información relacionada en unidades de tiempo que le toma al prototipo de aplicación al ejecutar una operación (insertar, actualizar, eliminar, listar), este parámetro sirve para medir el desempeño del motor de bases de datos. Este parámetro se mide en mili segundos. |
| CPU unidades de procesamiento | Proporciona información relacionada con las unidades de procesador que es utilizado por el prototipo de aplicación al ejecutar una operación (Insertar, actualizar, eliminar, listar). Se mide en unidades de procesamiento |
| Memoria RAM | Proporciona información relacionada a la cantidad de memoria utilizada por el prototipo de aplicación en la ejecución de una operación (Insertar, actualizar, eliminar, listar). Se mide en bytes. |
| Almacenamiento | Proporciona la información relacionada al espacio de almacenamiento utilizado del archivo de base de datos en la ejecución de una operación (Insertar, actualizar, eliminar, listar). Se mide en bytes. |

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

Una vez definido cada parámetro, se determinó un total de 4 indicadores (insertar, actualizar, eliminar, consultar), operaciones de las base de datos que permitirán obtener los datos necesarios en tiempo de ejecución de los prototipos para medir la eficiencia de los RDBMS ligeros en comparación.

## Ponderación de los parámetros de comparación

Para determinar la eficiencia en el consumo de recurso hardware de los RDBMS ligeros, se define una ponderación para dar un valor a cada uno de los parámetros que se comparan en esta investigación. Esta se realizó en base al criterio de los desarrolladores de esta investigación, tomando en consideración el nivel de importancia que cada uno representa en cuanto al utilizo de estos recursos al realizar las operaciones de insertar, actualizar, eliminar y listar con los prototipos desarrollados.

Se da un valor del 40% a la memoria RAM ya que este es un recurso aún limitado en los dispositivo móviles y de acuerdo a “límites y estándares que Microsoft impone para el correcto funcionamiento de una aplicación móvil para el SO Windows Phone no debe superar un top límite de MB” (App memory limits for Windows Phone*,* Microsoft 2015), de lo contrario la aplicación colapsaría, por esta razón se le da mayor peso a este recurso.

Por otra parte el 60% restante viene distribuido de la siguiente manera: el 25% para CPU unidades de tiempo, 25% para CPU unidades de procesamiento, y el 10% para el almacenamiento, ya que la tecnología en estos tiempos para este tipo de recursos ha mejorado mucho en los dispositivos móviles. En la Tabla 2‑2 se indica la ponderación para los parámetros en comparación:

Tabla 2‑2: Ponderación de los parámetros

|  |  |
| --- | --- |
| PARÁMETROS | PONDERACIÓN |
| Uso RAM | 40% |
| Uso CPU unidades de tiempo | 25% |
| Uso CPU unidades procesamiento | 25% |
| Almacenamiento | 10% |
| TOTAL | **100%** |

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

## Diseño del experimento

Para comparar la eficiencia entre los dos RDMBS ligeros se determinaron funcionalidades básicas necesarias de las bases de datos tales como insertar, actualizar, eliminar, consultar; creando prototipos de una aplicación móvil tanto para SQLite como para SQL Server Compact en la plataforma Windows Phone, los cuales se indican en el anexo A.

Definidas estas funcionalidades se procedió a la instalación de las herramientas tales como:

**Visual Studio 2013** **con el SDK de Windows Phone,** IDE de desarrollo utilizado para la construcción de los prototipos por cada RDBMS ligero y sus ejecuciones por medio del emulador de Windows Phone 8.1 WVGA, a su vez monitorear los puntos (marcas) de utilización del CPU en unidades de tiempo y unidades de procesamiento a través de la herramienta Profiling Tools, con la API CLASS DeviceStatusy la propiedad ApplicationPeakMemoryUsage se obtuvo la información del dispositivo en el cual se está ejecutando el prototipo y el pico máximo de memoria utilizado, con CLASS FileInfo se obtuvo el tamaño en bytes de la bases de datos.

**IsoStoreSpy,** explorador de archivos que permitió observar y extraer los archivos generados por cada ejecución de los prototipos con los datos capturados para posteriormente ser analizados.

Para el procesamiento de los datos cuantitativos obtenidos se utilizó el software estadístico SPSS de IBM, logrando así realizar la estadística descriptiva e inferencial.

Gracias a la potencialidad de estas herramientas se logró consolidar una gran cantidad de datos para su respectivo análisis.

### Cálculo de la muestra

Para el cálculo de muestra poblacional debido a que no se conoce el universo de estudio, se estableció el valor de desviación estándar en 0.05 que equivale al 5% del margen de error, un nivel de confianza de 1.96 que equivale al 95% de veracidad y a través de la fórmula siguiente se calculó el tamaño de la muestra poblacional:

Donde:

n = Tamaño de la muestra

Z = 1,96 que es el nivel de confianza del 95%

p = Probabilidad de que suceda el evento (0.5).

q = 1-p.

d = margen de error admitido.

Entonces:

Aplicando la fórmula mencionada se obtienen un valor de 384 muestras para el experimento.

### Descripción del ambiente de pruebas

Las características del computador que se utilizó para la ejecución del emulador mencionado con los prototipos son las que indica la tabla siguiente:

Tabla 2‑3: Especificación hardware del equipo

|  |  |
| --- | --- |
| HARDWARE | CARACTERÍSTICAS |
| Disco | SSD (250 GB) |
| RAM | DDR 3 (8 GB) |
| CPU | Intel Core i7 2.20 GHZ |

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

### Prototipos para las pruebas

Con la finalidad de que las pruebas se realicen en un entorno de total igualdad para SQLite y SQL Server Compact se diseñaron 8 prototipos con funcionalidades exactas 4 por cada uno (insertar, modificar, eliminar, y listar) los cuales se muestran en la tabla 2-4:

Tabla 2‑4: Prototipos para las pruebas

|  |  |
| --- | --- |
| SQLite | SQL Server Compact Edition |
| Prototipo INSERT | **Prototipo INSERT** |
| Prototipo UPDATE | **Prototipo UPDATE** |
| Prototipo DELETE | **Prototipo DELETE** |
| Prototipo SELECT | **Prototipo SELECT** |

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

Cada prototipo fue creado y desarrollado por los autores de esta investigación obteniendo 384 muestras por cada uno de ellos con inserciones, eliminaciones, actualizaciones, consultas por cada RDBMS como indican las tablas 2-5 y 2-6, esto para forzar a la aplicación y poder observar que tanto consume los recursos hardware al momento de su ejecución al realizar estas operaciones.

Tabla 2‑5: Cantidad de datos obtenidos prototipos SQLite

|  |  |
| --- | --- |
| PROTOTIPOS SQLITE | # MUESTRAS |
| INSERT | 384 |
| UPDATE | 384 |
| DELETE | 384 |
| SELECT | 384 |
| TOTAL | **1536** |

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

Tabla 2‑6: Cantidad de datos obtenidos prototipos SQL Server Compact Edition

|  |  |
| --- | --- |
| PROTOTIPOS SQL SERVER COMAPCT | # MUESTRAS |
| INSERT | 384 |
| UPDATE | 384 |
| DELETE | 384 |
| SELECT | 384 |
| TOTAL | **1536** |

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

# CAPÍTULO III

# RESULTADOS, DISCUSIÓN Y ANÁLISIS Y DE RESULTADOS

El análisis de resultado de datos es una de las partes más importantes en esta investigación, con el empleo de métodos estadísticos por medio del software SPSS de IBM tales como la estadística descriptiva e inferencial, se analizará los resultados obtenidos con las pruebas realizadas a través de los prototipos desarrollados y así obtener la ayuda necesaria para la toma de decisiones en cuanto a la demostración de la hipótesis planteada al inicio de esta investigación.

## Análisis preliminar de datos

Para determinar la prueba de inferencia estadística para el test de la hipótesis, se tomará en cuenta las siguientes suposiciones:

* Que los datos se distribuyan normalmente (prueba de normalidad).
* Que las muestras sean independientes.

Para ello se define inicialmente a la prueba de normalidad como “Pruebas para determinar si un conjunto de datos se distribuyen de una manera que es consistente con una distribución normal”.

Se determinó en esta investigación un valor de significancia (α) del 5% equivalente al 0.05, dicho valor es establecido por defecto para la prueba de hipótesis, siendo así el 95% para el nivel de confianza. Sucesivamente se realizó la prueba de normalidad Kolmogorov – Smirnov por cada uno de los prototipos desarrollados (Insertar, Eliminar, Actualizar, Consultas), ya que el tamaño de la muestra (384) es mayor a 50 para dos grupos de datos, en este caso por cada RDBMS ligero SQLite y SQL Server Compact.

En la tabla 3-1, haciendo uso del software estadístico SPSS de IBM se puede observar la prueba de normalidad aplicada al prototipo Insert para determinar si los datos tienen una distribución normal:

Tabla 3‑1: Prueba de normalidad prototipo Insert

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PARÁMETRO DE COMPARACIÓN | RDBMS LIGERO | Kolmogorov - Smirnov | | |
|  |  | **Estadístico** | **gl** | **P-valor/Sig.** |
| Unidades de tiempo Uso CPU | SQLite | 0,242 | 384 | 0,000 |
| SQL Server Compact | 0,176 | 384 | 0,000 |
| Unidades de procesamiento uso CPU | SQLite | 0,118 | 384 | 0,000 |
| SQL Server Compact | 0,151 | 384 | 0,000 |
| Pico máximo de uso de memoria | SQLite | 0,377 | 384 | 0,000 |
| SQL Server Compact | 0,443 | 384 | 0,000 |
| Almacenamiento | SQLite | 209920 | 384 | 0,000 |
| SQL Server Compact | 327680 | 384 | 0,000 |

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

En la tabla anterior se observa que el nivel de significancia o P-Valor (Columna Sig.) para cada RDBMS es menor a 0.05 (valor del α), por ende los datos no están distribuidos normalmente en cada una de la muestras independientes, descartando así la utilización de la prueba paramétrica T de Student, a su vez se utilizará la prueba no paramétrica U de Mann-Whitney enfocada al estudio entre dos grupos de datos (en el caso de esta investigación SQLite y SQL Server CE).

En la tabla 3-2, haciendo uso del software estadístico SPSS de IBM se puede observar la prueba de normalidad aplicada al prototipo Update para determinar si los datos tienen una distribución normal:

Tabla 3‑2: Prueba de normalidad prototipo Update

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PARÁMETRO | RDBMS LIGERO | Kolmogorov - Smirnov | | |
|  |  | **Estadístico** | **gl** | **P-valor/Sig.** |
| Unidades de tiempo Uso CPU | SQLite | 0,412 | 384 | 0,000 |
| SQL Server Compact | 0,276 | 384 | 0,000 |
| Unidades de procesamiento uso CPU | SQLite | 0,289 | 384 | 0,000 |
| SQL Server Compact | 0,171 | 384 | 0,000 |
| Pico máximo de uso de memoria | SQLite | 0,437 | 384 | 0,000 |
| SQL Server Compact | 0,395 | 384 | 0,000 |
| Almacenamiento | SQLite | 209920 | 384 | 0,000 |
| SQL Server Compact | 327680 | 384 | 0,000 |

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

Se observa en la tabla que el nivel de significancia o P-Valor (Columna Sig.) para cada RDBMS es menor a 0.05 (valor del α), por ende los datos no están distribuidos normalmente en cada una de la muestras independientes, descartando así la utilización de la prueba paramétrica T de Student, a su vez se utilizará la prueba no paramétrica U de Mann-Whitney enfocada al estudio entre dos grupos de datos (en el caso de esta investigación SQLite y SQL Server CE).

En la tabla 3-3, haciendo uso del software estadístico SPSS de IBM se puede observar la prueba de normalidad aplicada al prototipo Delete para determinar si los datos tienen una distribución normal:

Tabla 3‑3: Prueba de normalidad prototipo Delete

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PARÁMETRO DE COMPARACIÓN | RDBMS LIGERO | Kolmogorov - Smirnov | | |
|  |  | **Estadístico** | **gl** | **P-valor/Sig.** |
| Unidades de tiempo Uso CPU | SQLite | 0,352 | 384 | 0,000 |
| SQL Server Compact | 0,212 | 384 | 0,000 |
| Unidades de procesamiento uso CPU | SQLite | 0,133 | 384 | 0,000 |
| SQL Server Compact | 0,108 | 384 | 0,000 |
| Pico máximo de uso de memoria | SQLite | 0,424 | 384 | 0,000 |
| SQL Server Compact | 0,378 | 384 | 0,000 |
| Almacenamiento | SQLite | 209920 | 384 | 0,000 |
| SQL Server Compact | 327680 | 384 | 0,000 |

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

Se observa en la tabla que el nivel de significancia o P-Valor (Columna Sig.) para cada RDBMS es menor a 0.05 (valor del α), por ende los datos no están distribuidos normalmente en cada una de la muestras independientes, descartando así la utilización de la prueba paramétrica T de Student, a su vez se utilizará la prueba no paramétrica U de Mann-Whitney enfocada al estudio entre dos grupos de datos (en el caso de esta investigación SQLite y SQL Server CE).

En la tabla 3-4, haciendo uso del software estadístico SPSS de IBM se puede observar la prueba de normalidad aplicada al prototipo Select para determinar si los datos tienen una distribución normal:

Tabla 3‑4: Prueba de normalidad prototipo Select.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PARÁMETRO DE COMPARACIÓN | RDBMS LIGERO | Kolmogorov - Smirnov | | |
|  |  | **Estadístico** | **gl** | **P-valor/Sig.** |
| Unidades de tiempo Uso CPU | SQLite | 0,440 | 384 | 0,000 |
| SQL Server Compact | 0,068 | 384 | 0,000 |
| Unidades de procesamiento uso CPU | SQLite | 0,149 | 384 | 0,000 |
| SQL Server Compact | 0,099 | 384 | 0,000 |
| Pico máximo de uso de memoria | SQLite | 0,419 | 384 | 0,000 |
| SQL Server Compact | 0,372 | 384 | 0,000 |
| Almacenamiento | SQLite | 209920 | 384 | 0,000 |
| SQL Server Compact | 327680 | 384 | 0,000 |

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

Se observa en la tabla que el nivel de significancia o P-Valor (Columna Sig.) para cada RDBMS es menor a 0.05 (valor del α), por ende los datos no están distribuidos normalmente en cada una de la muestras independientes, descartando así la utilización de la prueba paramétrica T de Student, a su vez se utilizará la prueba no paramétrica U de Mann-Whitney enfocada al estudio entre dos grupos de datos (en el caso de esta investigación SQLite y SQL Server CE).

Para realizar inferencias estadísticas se debe adoptar un modelo de decisión para la determinación de las diferencias significativas por medio de la hipótesis nula o alterna. La estadística descriptiva se utilizará para recopilar, e interpretar los datos recolectados, de manera tal que se describa de forma fácil y rápida dicha información.

La estadística inferencial en cambio se utilizará para la toma de decisiones en base a los resultados, probando así si se cumple o no la hipótesis planteada.

## Estudio Comparativo

La medición de la eficiencia en el consumo de recursos hardware entre los RDBMS ligeros se la realizó por medio de las ejecuciones de los prototipos mencionados anteriormente. Para el RDBMS ligero que menos hace uso de estos recursos se toma en consideración la ponderación de la tabla 2-2 la cual asigna el 40%, 25%, 25% y 10% respectivamente por cada recurso hardware utilizado, mientras que para el RDBMS que utiliza más estos recursos, se le asigna el valor de ponderación aplicando una regla de tres inversa.

Para determinar el RDBMS ligero de mejor prestaciones en cuanto a la eficiencia, se realizaron pruebas con 384 muestras concurrentes con cada uno de los prototipos desarrollados, los resultados obtenidos fueron analizados y evaluados por los autores de esta investigación.

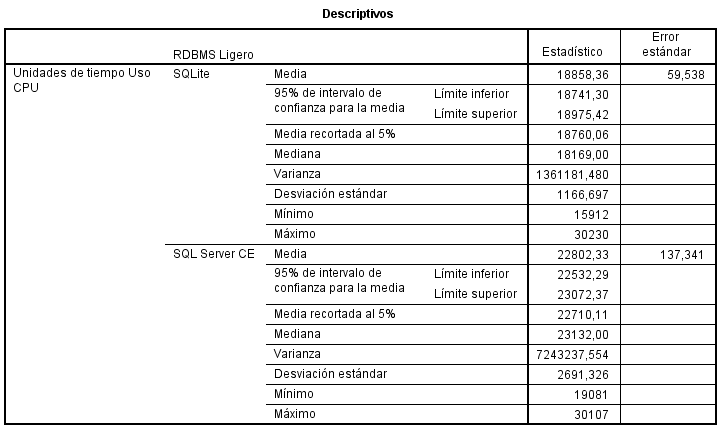
Los datos y resultados obtenidos para cada parámetro en comparación se indican a continuación.

### Eficiencia en Insert

#### **Uso CPU unidades de tiempo**

Se analiza la cantidad de uso del recurso CPU unidades de tiempo que emplea el prototipo insertar tanto para SQLite como para SQL Server Compact Edition al momento de su ejecución, en la Tabla 3‑5 se indica la estadística descriptiva con los resultados obtenidos con el software estadístico SPSS:

Tabla 3‑5: Estadística descriptiva del uso CPU unidades de tiempo - Insert



**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

Para SQLite se tiene lo siguiente:

La cantidad de uso del CPU en unidades de tiempo para este RDBMS ligero es de 18858,36 mili segundos con tendencia a variar ya sea por encima o por debajo de este valor con 59,538 mili segundos. De entre los valores obtenidos existe dispersión de los datos con respecto al promedio de uso de este recurso con 1166,697 mili segundos con valores extremos entre 15912 mili segundos y 30230 mili segundos.

Para SQL Server Compact Edition se tiene:

La cantidad de uso del CPU en unidades de tiempo para este RDBMS ligero es de 22802,33 mili segundos con una tendencia a variar ya sea por encima o por debajo de este valor con 137,341 mili segundos. De entre los valores obtenidos existe una mayor dispersión de los datos con respecto al promedio de uso de este recurso con 2691,326 mili segundos con valores extremos entre 19081 mili segundos y 30107 mili segundos.

Para el análisis inferencial de este parámetro se tiene:

**H0**: No existe una diferencia significativa de las unidades de tiempo medio de uso del CPU al insertar datos con SQLite con respecto a SQL Server Compact Edition.

**H1**: Existe una diferencia significativa de las unidades de tiempo medio de uso del CPU al insertar datos con SQLite con respecto a SQL Server Compact

Con la utilización del software estadístico SPSS en la tabla 3-6 se observa la prueba U de Mann Whitney para muestras independientes con los datos obtenidos:

Tabla 3‑6: Estadísticos de contraste uso CPU unidades de tiempo - prototipo Insert

|  |  |
| --- | --- |
|  | Uso CPU unidades de tiempo |
| U-Mann Whitney | 138213,000 mili segundos |
| Significación asintótica (Prueba bilateral) | 0,000 |

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

Mediante esta prueba no paramétrica se rechaza la hipótesis nula H0 y se acepta la alternativa H1, lo que implica que existe una diferencia significativa de las unidades de tiempo medio de uso de CPU entre las categorías de RDBMS ligeros debido a que el P-valor (significación asintótica) es menor que el alpha (0,000<0.05).

De acuerdo a la ponderación establecida en la Tabla 2‑2, para uso CPU unidades de tiempo se asigna al RDBMS ligero que satisface las condiciones mencionadas en dicho capítulo con el 25% de los valores y al RDBMS que no satisface se calcula el valor proporcional con una regla de tres inversa como se indica en la tabla 3-7:

Tabla 3‑7: Ponderación parámetro CPU unidades de tiempo - prototipo Insert

|  |  |  |
| --- | --- | --- |
| PROTIPO | RDBMS | VALOR DE PONDERACION |
| INSERT | SQLite | 25 % |
| SQL Server Compact Edition | 20,67 % |

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

En la Figura 3‑1 se observa la diferencia de promedio de uso CPU unidades de tiempo obtenido al ejecutar el prototipo Insert para los RDBMS ligeros SQLite y SQL Server CE.

Figura 3‑1: Promedio de uso CPU unidades de tiempo (ms) - prototipo Insert

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

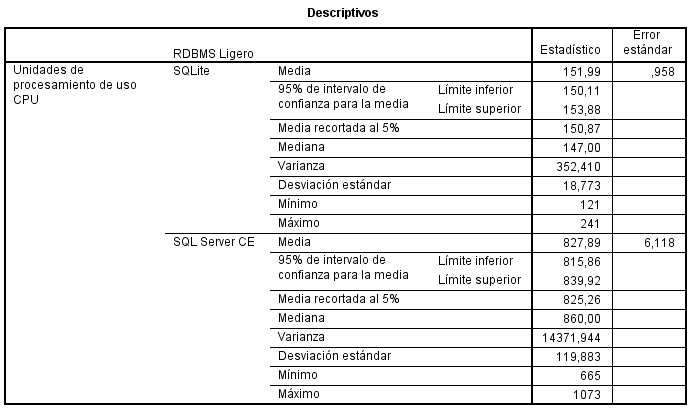
#### **Interpretación de resultados**

El RDBMS con menores unidades de tiempo de uso CPU es SQLite con un tiempo medio de 18858.36 ms (mili segundos) a diferencia de SQL Server CE con 22802,33ms (mili segundos). Conforme a la conclusión estadística de la prueba no paramétrica U de Mann-Whitney de la Tabla 3‑6, se comprueba que SQLite presenta diferencias significativas respecto a SQL Server Compact, además en la Tabla 3‑7 tiene un valor más alto en la ponderación con respecto al otro, ya que su tiempo promedio en el ingreso de datos es menor.

#### **Uso CPU unidades de procesamiento**

Se analiza la cantidad de uso del recurso CPU unidades de procesamiento que emplea el prototipo insertar tanto para SQLite como para SQL Server CE al momento de su ejecución, la Tabla 3‑8 indica la estadística descriptiva con los resultados obtenidos con el software estadístico SPSS:

Tabla 3‑8: Estadística Descriptiva uso CPU unidades de procesamiento - prototipo Insert



**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

Para SQLite se tiene lo siguiente:

La cantidad de uso en unidades de procesamiento del CPU para este RDBMS ligero es de 151,99 unidades de procesamiento posee una tendencia a variar ya sea por encima o por debajo de este valor con 0,958 unidades de procesamiento. De entre los valores obtenidos existe una menor dispersión de los datos con respecto al promedio de uso de este recurso con 18,773 unidades de procesamiento con valores extremos entre 121 unidades de procesamiento y 241 unidades de procesamiento.

Para SQL Server Compact Edition se tiene:

La cantidad de uso en unidades de procesamiento del CPU para este RDBMS ligero es de 827,89 unidades de procesamiento con una tendencia a variar ya sea por encima o por debajo de este valor con 6,118 unidades de procesamiento. De entre los valores obtenidos existe una mayor dispersión de los datos con respecto al promedio de uso de este recurso con 119,883 unidades de procesamiento con valores extremos entre 665 unidades de procesamiento y 1073 unidades de procesamiento.

Para el análisis inferencial de este parámetro se tiene:

**H0**: No existe una diferencia significativa del uso CPU unidades de procesamiento al insertar datos con SQLite con respecto a SQL Server Compact Edition.

**H1**: Existe una diferencia significativa del uso CPU unidades de procesamiento al Insertar datos con SQLite con respecto a SQL Server Compact.

Con el uso del software estadístico SPSS en la tabla 3-9 se observa la prueba U de Mann Whitney para muestras independientes con los datos obtenidos:

Tabla 3‑9: Estadísticos de contraste uso CPU unidades de procesamiento - prototipo Insert

|  |  |
| --- | --- |
|  | Uso CPU unidades de procesamiento |
| U-Mann Whitney | 147456,000 unidades de procesamiento |
| Significación asintótica (Prueba bilateral) | 0,000 |

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

Mediante esta prueba no paramétrica se rechaza la hipótesis nula H0 y se acepta la alternativa H1, lo que significa que si existe una diferencia significativa de uso del CPU unidades procesamiento entre las categorías de RDBMS ligeros debido a que el P-valor (Significación asintótica) es menor que el alpha (0,000<0.05).

De acuerdo a la ponderación establecida en la Tabla 2‑2, para uso CPU unidades de procesamiento, se asigna al RDBMS ligero que satisface las condiciones mencionadas en dicho capítulo con el 25% de los valores y al RDBMS que no satisface se calcula el valor proporcional con una regla de tres inversa con como se indica en la tabla 3-10:

Tabla 3‑10: Ponderación parámetro uso CPU unidades de procesamiento - prototipo Insert

|  |  |  |
| --- | --- | --- |
| PROTIPO | RDBMS | VALOR DE PONDERACION |
| INSERTAR | SQLite | 25 % |
| SQL Server Compact Edition | 4,58 % |

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

En la Figura 3‑2 se observa la diferencia de promedios del uso CPU unidades de procesamiento obtenidos con el prototipo Insert para los RDBMS ligeros SQLite y SQL Server Compact:

Figura 3‑2: Uso CPU unidades de procesamiento - prototipo Insert

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

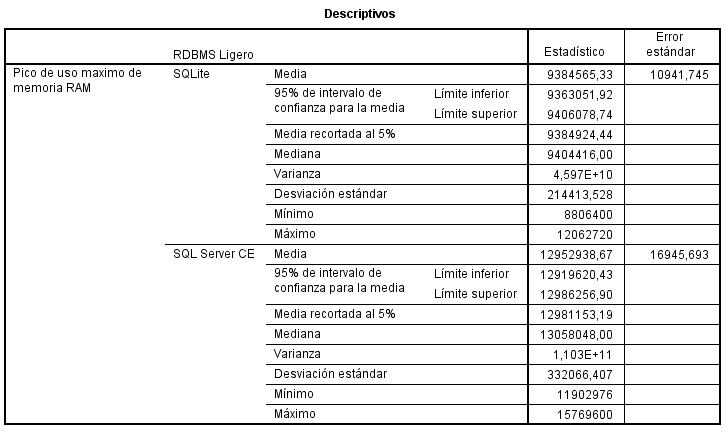
#### **Interpretación de resultados**

El RDBMS ligero con menores unidades de procesamiento es SQLite con una cantidad de trabajo medio de 157,99 unidades de procesamiento a diferencia de SQL Server Compact Edition con 827,89 unidades de procesamiento utilizadas. Conforme a la conclusión estadística de la prueba no paramétrica U de Mann-Whitney de la Tabla 3‑9 se comprueba que SQLite presenta diferencias significativas respecto a SQL Server Compact, en la Tabla 3‑10 tiene un valor más alto en la ponderación con respecto al otro, ya que las unidades de procesamiento utilizadas en el ingreso de datos son menor.

#### **Uso de RAM**

Se analiza la cantidad de uso del recurso RAM que emplea la ejecución del prototipo insertar tanto para SQLite como para SQL Server CE, la Tabla 3‑11 indica la estadística descriptiva con los resultados obtenidos para el uso de RAM con el software estadístico SPSS:

Tabla 3‑11: Estadística Descriptiva uso RAM - prototipo Insert



**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

Para SQLite se tiene lo siguiente:

El pico de uso máximo RAM para este RDBMS ligero es de 9384565,33 bytes, posee una tendencia a variar ya sea por encima o por debajo de este valor con 10941,745 bytes. De entre los valores obtenidos existe dispersión de los datos con respecto al promedio de uso de este recurso con 214413,528 bytes de con valores extremos entre 8806400 bytes y 12062720 bytes.

Para SQL Server Compact Edition se tiene:

El pico de uso máximo RAM para este RDBMS ligero es de 12952938,67 bytes con una tendencia a variar ya sea por encima o por debajo de este valor con 16945,693 bytes. De entre los valores obtenidos existe una mayor dispersión de los datos con respecto al promedio de uso de este recurso con 332066,407 bytes con valores extremos entre 11902976 bytes y 15769600 bytes.

Para el análisis inferencial de este parámetro se tiene:

**H0**: No existe una diferencia significativa del uso de RAM al insertar datos con SQLite con respecto a SQL Server Compact Edition.

**H1**: Existe una diferencia significativa del uso de RAM al insertar datos con SQLite con respecto a SQL Server Compact.

Con la utilización del software estadístico SPSS se hace el análisis de los datos, la tabla 3-12 indica la prueba U de Mann Whitney para muestras independientes con los datos obtenidos:

Tabla 3‑12: Estadísticos de contraste uso RAM - prototipo Insert

|  |  |
| --- | --- |
|  | Uso RAM |
| U-Mann Whitney | 147455,000 bytes |
| Significación asintótica (Prueba bilateral) | 0,000 |

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

Mediante esta prueba no paramétrica se rechaza la hipótesis nula H0 y se acepta la alternativa H1, indicando que existe diferencia significativa de uso RAM entre las categorías de RDBMS ligeros debido a que el P-valor (Significación asintótica) es menor que el alpha (0,000<0.05).

De acuerdo a la ponderación establecida en la Tabla 2‑2 para uso de RAM, se asigna al RDBMS ligero que satisface las condiciones mencionadas en dicho capítulo con el 40% de los valores y al RDBMS que no satisface se calcula el valor proporcional con una regla de tres inversa con como se indica en la tabla 3-13:

Tabla 3‑13: Ponderación parámetro uso RAM - prototipo Insert

|  |  |  |
| --- | --- | --- |
| PROTIPO | RDBMS | VALOR DE PONDERACION |
| INSERT | SQLite | 40 % |
| SQL Server Compact Edition | 18,11 % |

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

La Figura 3‑3 indica la diferencia de promedios del uso RAM obtenidos con el prototipo INSERTAR para los RDBMS ligeros SQLite y SQL Server Compact Edition:

Figura 3‑3: Promedio en bytes del uso del recurso RAM - prototipo Insert

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

#### **Interpretación de resultados**

El RDBMS ligero con menor uso de este recurso en la ejecución de este prototipo es SQLite con una media de 9384565,33 bytes a diferencia de SQL Server Compact Edition con 12952938,66 bytes. Conforme a la conclusión estadística de la prueba no paramétrica U de Mann-Whitney, se comprueba que SQLite presenta diferencias significativas en el uso RAM respecto a SQL Server Compact, además la Tabla 3‑13 indica que tiene un valor más alto en la ponderación con respecto al otro, ya que de acuerdo a la media del consumo de este recurso en el ingreso de datos es menor, por lo tanto SQLite presenta una mayor eficiencia.

#### **Almacenamiento**

Se analiza el uso del recurso Almacenamiento que emplea el prototipo insertar tanto para SQLite como para SQL Server CE al momento su ejecución, el valor del archivo de almacenamiento no varía tanto para él un RDBMS como para el otro, es un valor constante al realizar las inserciones con la generación de datos aleatorios, la información obtenida para este parámetro nos representativa, por ende se omite la inferencia estadística y se toma en consideración el de menor tamaño de archivo.

La tabla 3-14 indica los resultados obtenidos para el uso de este recurso hardware:

Tabla 3‑14: Tamaños de archivo Almacenamiento - prototipo Insert

|  |  |
| --- | --- |
| RDBMS | TAMAÑO DE ARCHIVO UTILIZADO |
| SQLite | 209920 bytes |
| SQL Server CE | 327680 bytes |

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

De los valores obtenidos del archivo de almacenamiento al momento de realizar las inserciones por cada RDBMS, SQLite es de menor valor con relación a SQL server Compact en el tamaño de archivo utilizado.

De acuerdo a la ponderación establecida en la Tabla 2‑2 para el uso de Almacenamiento, se asigna al RDBMS ligero que satisface las condiciones mencionadas en dicho capítulo con el 10% de los valores y al RDBMS que no satisface se le calcula el valor proporcional con una regla de tres inversa como indica la tabla 3-15:

Tabla 3‑15: Ponderación parámetro uso Almacenamiento - prototipo Insert

|  |  |  |
| --- | --- | --- |
| PROTIPO | RDBMS | VALOR DE PONDERACION |
| INSERT | SQLite | 10 % |
| SQL Server Compact Edition | 6,4 % |

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

La figura 3-4 indica el promedio de uso Almacenamiento obtenidos con el prototipo Insertar para los RDBMS ligeros SQLite y SQL Server Compact Edition:

Figura 3‑4: Promedio del uso Almacenamiento - prototipo Insert

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

#### **Interpretación de resultados**

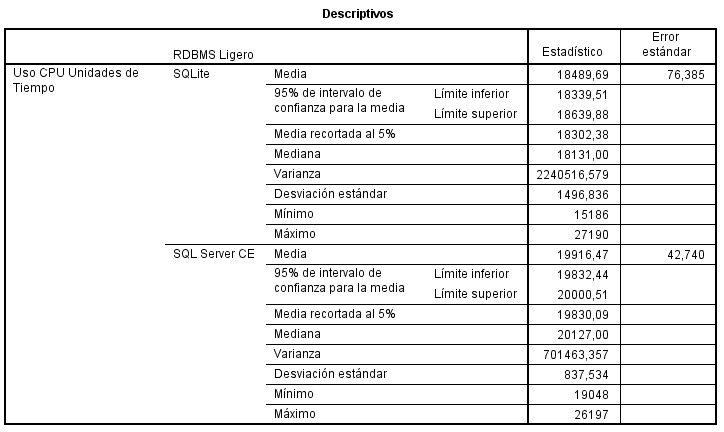
El RDBMS ligero con menor uso del recurso almacenamiento en la ejecución del prototipo insertar es SQLite con un valor de 209920 bytes como tamaño de archivo a diferencia de SQL Server Compact Edition con 327680 bytes, por lo tanto presenta una mayor eficiencia.

### Eficiencia en Update

#### **Uso CPU unidades de tiempo**

Se analiza la cantidad de uso del recurso CPU unidades de tiempo que emplea el prototipo Update tanto para SQLite como para SQL Server CE al momento de su ejecución, la **Tabla 3‑16** indica la estadística descriptiva con los resultados obtenidos con el software estadístico SPSS:

Tabla 3‑16: Estadística Descriptiva uso CPU unidades de tiempo - prototipo Update



**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

Para SQLite se tiene lo siguiente:

La cantidad promedio de uso en unidades de tiempo del CPU para este RDBMS ligero es de 18489,69 mili segundos con tendencia a variar ya sea por encima o por debajo de este valor con 76,385 mili segundos. De entre los valores obtenidos existe dispersión de los datos con respecto al promedio de uso de este recurso con 1496,836 mili segundos con valores extremos entre 15186 mili segundos y 27190 mili segundos.

Para SQL Server CE se tiene:

La cantidad de uso del CPU en unidades de tiempo para este RDBMS ligero es de 19916,47 mili segundos con tendencia a variar ya sea por encima o por debajo de este valor con 42,740 mili segundos. De entre los valores obtenidos existe una mayor dispersión de los datos con respecto al promedio de uso de este recurso con 837,534 mili segundos con valores extremos entre 19048 mili segundos y 26197 mili segundos.

Para el análisis inferencial de este parámetro se tiene:

**H0**: No existe una diferencia significativa de las unidades de tiempo medio de uso del CPU al actualizar datos con SQLite con respecto a SQL Server Compact Edition.

**H1**: Existe una diferencia significativa de las unidades de tiempo medio de uso del CPU al actualizar datos con SQLite con respecto a SQL Server Compact.

Con la utilización del software estadístico SPSS, la tabla3-17 indica la prueba U de Mann Whitney para muestras independientes con los datos obtenidos:

Tabla 3‑17: Estadísticos de contraste uso CPU unidades de tiempo - prototipo Update

|  |  |
| --- | --- |
|  | Uso CPU unidades de tiempo |
| U-Mann Whitney | 130582,000 mili segundos |
| Significación asintótica (Prueba bilateral) | 0,000 |

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

Mediante esta prueba no paramétrica se rechaza la hipótesis nula H0 y se acepta la alternativa H1, indica que si existe una diferencia significativa de las unidades de tiempo medio de uso de CPU entre las categorías de RDBMS ligeros debido a que el P-valor (significación asintótica) es menor que el alpha (0,000<0.05).

De acuerdo a la ponderación establecida en la Tabla 2‑2, para uso CPU unidades de tiempo se asigna al RDBMS ligero que satisface las condiciones mencionadas en dicho capítulo con el 25% de los valores y al RDBMS que no satisface se calcula el valor proporcional con una regla de tres inversa con como se indica la tabla 3-18:

Tabla 3‑18: Ponderación parámetro CPU unidades de tiempo - Update

|  |  |  |
| --- | --- | --- |
| PROTIPO | RDBMS | VALOR DE PONDERACION |
| ACTUALIZAR | SQLite | 25 % |
| SQL Server Compact Edition | 23,20 % |

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

La figura 3-5 indica el promedio de uso del CPU unidades de tiempo obtenidos con el prototipo UPDATE:

Figura 3‑5: Promedio de uso CPU unidades de tiempo (ms) - prototipo Update

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

#### **Interpretación de resultados**

El RDBMS ligero con menores unidades de tiempo de uso CPU en la ejecución del prototipo es SQLite con un tiempo medio de 18489,69 mili segundos a diferencia de SQL Server Compact Edition con 19916,47 mili segundos. Conforme la prueba no paramétrica de hipótesis utilizando U de Mann-Whitney, se comprueba que SQLite presenta diferencias respecto a SQL Server Compact, la Tabla 3‑18 indica que tiene un valor más alto en la ponderación con respecto al otro, ya que su tiempo promedio en el ingreso de datos es menor, por lo tanto SQLite presenta una mayor eficiencia en el consumo de este recurso.

#### Uso CPU unidades de procesamiento

Se analiza la cantidad de uso del recurso CPU unidades de procesamiento que emplea el prototipo actualizar tanto para SQLite como para SQL Server CE al momento de su ejecución, la Tabla 3-19, indica la estadística descriptiva con los resultados obtenidos con el software estadístico SPSS:

|  |
| --- |
| Tabla 3‑19: Estadística Descriptiva uso CPU unidades de procesamiento - prototipo Update    **FUENTE**: Darwin Cedeño y Diego Quinatoa 2015 |

Para SQLite se tiene lo siguiente:

La cantidad de uso en unidades de procesamiento del CPU para este RDBMS ligero es de 161,65 unidades de procesamiento; con una tendencia a variar ya sea por encima o por debajo de este valor con 1,485 unidades de procesamiento. De entre los valores obtenidos existe una menor dispersión de los datos con respecto al promedio de uso de este recurso con 29,102 unidades de procesamiento con valores extremos entre 118 unidades de procesamiento y 292 unidades de procesamiento.

Para SQL Server CE se tiene:

La cantidad de uso en unidades de procesamiento del CPU para este RDBMS ligero es de 798,82 unidades de procesamiento con una tendencia a variar ya sea por encima o por debajo de este valor con 1,605 unidades de procesamiento. De entre los valores obtenidos existe una dispersión de los datos con respecto al promedio de uso de este recurso con 31,450 unidades de procesamiento valores extremos entre 740 unidades de procesamiento y 1056 unidades de procesamiento

Para el análisis inferencial de este parámetro se tiene:

**H0**: No existe una diferencia significativa del uso CPU unidades de procesamiento al actualizar datos con SQLite con respecto a SQL Server Compact Edition.

**H1**: Existe una diferencia significativa del uso CPU unidades de procesamiento en la operación actualizar de SQLite con respecto a SQL Server Compact.

Con la utilización del software estadístico SPSS, la tabla 3-20 indica la prueba U de Mann Whitney para muestras independientes con los datos obtenidos:

Tabla 3‑20: Estadísticos de contraste uso CPU unidades de procesamiento - prototipo Update

|  |  |
| --- | --- |
|  | Uso CPU unidades de procesamiento |
| U-Mann Whitney | 147456,000 unidades de procesamiento |
| Significación asintótica (Prueba bilateral) | 0,000 |

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

Mediante esta prueba no paramétrica se rechaza la hipótesis nula H0 y se acepta la alternativa H1, lo cual indica que si existe una diferencia significativa de uso CPU unidades de procesamiento entre las categorías de RDBMS ligeros debido a que el P-valor (Significación asintótica) es menor que el alpha (0,000<0.05).

De acuerdo a la ponderación establecida en la Tabla 2‑2 para uso CPU unidades de procesamiento, se asigna al RDBMS ligero que satisface las condiciones mencionadas en dicho capítulo con el 25% de los valores y al RDBMS que no satisface se calcula el valor proporcional con una regla de tres inversa con como indica la tabla 3-21:

Tabla 3‑21: Ponderación parámetro uso CPU unidades de procesamiento - Update

|  |  |  |
| --- | --- | --- |
| PROTIPO | RDBMS | VALOR DE PONDERACION |
| ACTUALIZAR | SQLite | 25 % |
| SQL Server Compact Edition | 5,06 % |

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

La figura 3-6 indica el promedio de uso CPU unidades de procesamiento obtenidos con el prototipo Update para los RDBMS ligeros SQLite y SQL Server Compact:

Figura 3‑6: Promedio de uso CPU unidades de procesamiento - Update

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

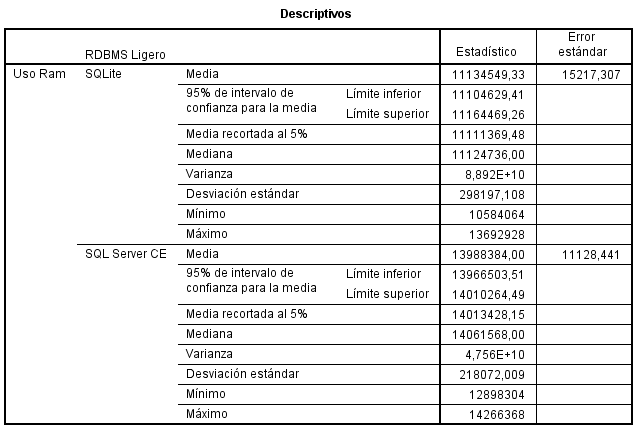
#### **Interpretación de resultados**

El RDBMS ligero con menores unidades de procesamiento en la ejecución del prototipo es SQLite con una cantidad de trabajo medio de 161,55 unidades de procesamiento a diferencia de SQL Server Compact Edition con 798,82 unidades de procesamiento utilizadas. Conforme la prueba no paramétrica de hipótesis U de Mann-Whitney, se comprueba que SQLite presenta diferencias en el uso CPU unidades de procesamiento respecto a SQL Server Compact, la Tabla 3‑21 indica que tiene un valor más alto en la ponderación con respecto al otro, ya que su promedio en el ingreso de datos es menor, por lo tanto SQLite presenta una mayor eficiencia en el consumo de este recurso.

#### **Uso de RAM**

Se analiza la cantidad de uso del recurso RAM que emplea la ejecución del prototipo actualizar tanto para SQLite como para SQL Server CE, la **Tabla 3-22** indica la estadística descriptiva con los resultados obtenidos para el uso de RAM con el software estadístico SPSS:

Tabla 3‑22: Estadística Descriptiva uso RAM - prototipo Update



**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

Para SQLite se tiene lo siguiente:

El pico de uso máximo RAM para este RDBMS ligero es de 11134549,33 bytes con tendencia a variar por encima o por debajo de este valor con 15217,307 bytes. De entre los valores obtenidos existe dispersión de los datos con respecto al promedio de uso de este recurso con 298197,108 bytes con valores extremos entre 10584064 bytes y 13692928 bytes.

Para SQL Server Compact Edition se tiene:

El pico de uso máximo RAM para este RDBMS ligero es de 13988384,00 bytes con tendencia a variar por encima o por debajo de este valor con 11128,441 bytes. De entre los valores obtenidos existe una dispersión de los datos con respecto al promedio de uso de este recurso con 218072,6 bytes con valores extremos entre 12898304 bytes y 14266368 bytes.

Para el análisis inferencial de este parámetro se tiene:

**H0**: No existe diferencia significativa del uso de RAM al actualizar datos con SQLite con respecto a SQL Server Compact Edition.

**H1**: Existe diferencia significativa del uso de RAM al actualizar datos con SQLite con respecto a SQL Server Compact.

Con el uso del software estadístico SPSS se hace el análisis de los datos, la tabla 3-23 indica la prueba U de Mann Whitney para muestras independientes con los datos obtenidos:

Tabla 3‑23: Estadísticos de contraste uso RAM - prototipo Update

|  |  |
| --- | --- |
|  | Uso RAM |
| U-Mann Whitney | 147192,000 |
| Significación asintótica (Prueba bilateral) | 0,000 |

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

Mediante esta prueba no paramétrica se rechaza la hipótesis nula H0 y se acepta la alternativa H1, lo cual indica que existe diferencia significativa de uso RAM entre las categorías de RDBMS ligeros debido a que el P-valor (significación asintótica) es menor que el alpha (0,000<0.05).

De acuerdo a la ponderación establecida en la Tabla 2‑2 para uso de RAM, se asigna al RDBMS ligero que satisface las condiciones mencionadas en dicho capítulo con el 40% de los valores y al RDBMS que no satisface se calcula el valor proporcional con una regla de tres inversa con como se indica la tabla 3-24:

Tabla 3‑24: Ponderación parámetro uso RAM - prototipo Update

|  |  |  |
| --- | --- | --- |
| PROTIPO | RDBMS | VALOR DE PONDERACION |
| ACTUALIZAR | SQLite | 40 % |
| SQL Server Compact Edition | 31,84 % |

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

La Figura 3‑7 indica el promedio de uso RAM obtenido con el prototipo Update para los RDBMS ligeros SQLite y SQL Server CE:

Figura 3‑7: Promedio en bytes del uso del recurso RAM - prototipo Update

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

#### **Interpretación de resultados**

El RDBMS ligero con menor uso de este recurso en la ejecución de este prototipo es SQLite con una media de 11134549, 33 bytes a diferencia de SQL Server Compact Edition con 13988384 bytes. Conforme a la prueba no paramétrica de hipótesis U de Mann-Whitney, se comprueba que SQLite presenta diferencias significativas en el uso RAM respecto a SQL Server Compact, la Tabla 3‑24 indica que tiene un valor más alto en la ponderación con respecto al otro, ya que de acuerdo a la media del consumo de este recurso al actualizar datos es menor, por lo tanto SQLite presenta una mayor eficiencia.

#### **Almacenamiento**

Se analiza el uso del recurso Almacenamiento que emplea el prototipo actualizar tanto para SQLite como para SQL Server CE al momento de su ejecución, el valor del archivo de almacenamiento no varía tanto para él un RDBMS como para el otro, es un valor constante al realizar las actualizaciones con la generación de datos aleatorios, la información obtenida para este parámetro no es representativa, por ende se toma en consideración el de menor tamaño de archivo.

La **Tabla 3‑25** indica los resultados obtenidos para el uso de este recurso hardware con el tamaño de archivo utilizado:

Tabla 3‑25: Tamaños de archivo Almacenamiento - prototipo Update

|  |  |
| --- | --- |
| RDBMS | TAMAÑO DE ARCHIVO UTILIZADO |
| SQLite | 209920 bytes |
| SQL Server CE | 327680 bytes |

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

De los valores obtenidos del archivo de almacenamiento al momento de realizar las actualizaciones por cada RDBMS ligero, SQLite es de menor valor con relación a SQL server Compact en el tamaño de archivo utilizado.

De acuerdo a la ponderación establecida en la Tabla 2‑2 para el uso de Almacenamiento, se asigna al RDBMS ligero que satisface las condiciones mencionadas en dicho capítulo con el 10% de los valores y al RDBMS que no satisface se le calcula el valor proporcional con una regla de tres inversa con como indica la Tabla 3-26:

Tabla 3‑26: Ponderación parámetro uso Almacenamiento - prototipo Update

|  |  |  |
| --- | --- | --- |
| PROTIPO | RDBMS | VALOR DE PONDERACION |
| ACTUALIZAR | SQLite | 10 % |
| SQL Server Compact Edition | 6,4 % |

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

La figura 3-8 indica el promedio de uso Almacenamiento obtenidos con el prototipo Update para los RDBMS ligeros SQLite y SQL Server Compact Edition:

Figura 3‑8: Promedio del uso Almacenamiento - prototipo Update

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

#### **Interpretación de resultados**

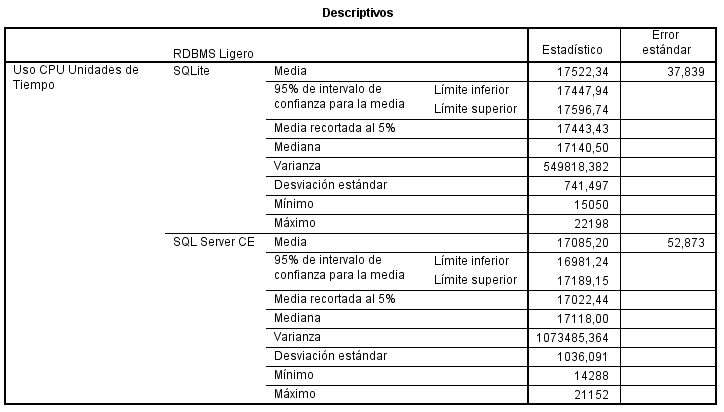
El RDBMS ligero con menor uso de este recurso en la ejecución de este prototipo es SQLite con un valor de 209920 bytes para el tamaño del archivo a diferencia de SQL Server Compact Edition con 327680 bytes, por lo tanto presenta una mayor eficiencia.

### Eficiencia en Delete

#### **Uso CPU unidades de tiempo**

Se analiza la cantidad de unidades de tiempo de uso del recurso CPU que emplea el prototipo Delete tanto para SQLite como para SQL Server CE al momento de su ejecución, la Tabla 3-27 indica la estadística descriptiva con los resultados obtenidos con el software estadístico SPSS:

Tabla 3‑27: Estadística Descriptiva uso CPU unidades de tiempo - prototipo Delete



**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

Para SQLite se tiene lo siguiente:

La cantidad de uso en unidades de tiempo del CPU para este RDBMS ligero es de 17522,34 mili segundos con tendencia a variar ya sea por encima o por debajo de este valor con 37,839. De entre los valores obtenidos existe dispersión de los datos con respecto al promedio de uso de este recurso con 741,497 mili segundos con valores extremos entre 15050 mili segundos y 22198 mili segundos.

Para SQL Server CE se tiene:

La cantidad de uso en unidades de tiempo del CPU para este RDBMS ligero es de 17085,20 mili segundos con tendencia a variar ya sea por encima o por debajo de este valor con 52,873 mili segundos. De entre los valores obtenidos existe mayor dispersión de los datos con respecto al promedio de uso de este recurso con 1036,091 mili segundos con valores extremos entre 14288 mili segundos y 21152 mili segundos.

Para el análisis inferencial de este parámetro se tiene:

**H0**: No existe una diferencia significativa de las unidades de tiempo medio de uso del CPU al eliminar datos con SQLite con respecto a SQL Server Compact Edition.

**H1**: Existe una diferencia significativa de las unidades de tiempo medio de uso del CPU al eliminar datos con SQLite con respecto a SQL Server Compact.

Con la utilización del software estadístico SPSS la Tabla 3-28 indica la prueba U de Mann Whitney para muestras independientes con los datos obtenidos:

Tabla 3‑28: Estadísticos de contraste uso CPU unidades de tiempo - prototipo Delete

|  |  |
| --- | --- |
|  | Uso CPU unidades de tiempo |
| U-Mann Whitney | 51298,000 mili segundos |
| Significación asintótica (Prueba bilateral) | 0,000 |

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

Mediante esta prueba no paramétrica se rechaza la hipótesis nula H0 y se acepta la alternativa H1, es decir que si existe una diferencia significativa de las unidades de tiempo medio de uso de CPU entre las categorías de RDBMS ligeros debido a que el P-valor (significación asintótica) es menor que el alpha (0,000<0.05).

De acuerdo a la ponderación establecida en la Tabla 2‑2, para uso CPU unidades de tiempo se asigna al RDBMS ligero que satisface las condiciones mencionadas en dicho capítulo con el 25% de los valores y al RDBMS que no satisface se calcula el valor proporcional con una regla de tres inversa como indica la Tabla 3-29:

Tabla 3‑29: Ponderación parámetro CPU unidades de tiempo - prototipo Delete

|  |  |  |
| --- | --- | --- |
| PROTIPO | RDBMS | VALOR DE PONDERACION |
| DELETE | SQLite | 24,37 % |
| SQL Server Compact Edition | 25 % |

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

La Figura 3-9 indica el promedio de uso del CPU unidades de tiempo obtenidos con el prototipo Delete para los RDBMS ligeros SQLite y SQL Server Compact Edition, en donde el RDBMS ligero con menor uso de unidades de tiempo de CPU para la ejecución del prototipo es SQL Server Compact Edition con un tiempo medio de 17085,20ms a diferencia de SQLite con 17522,34ms.

Figura 3‑9: Promedio de uso CPU unidades de tiempo (ms) – prototipo Delete

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

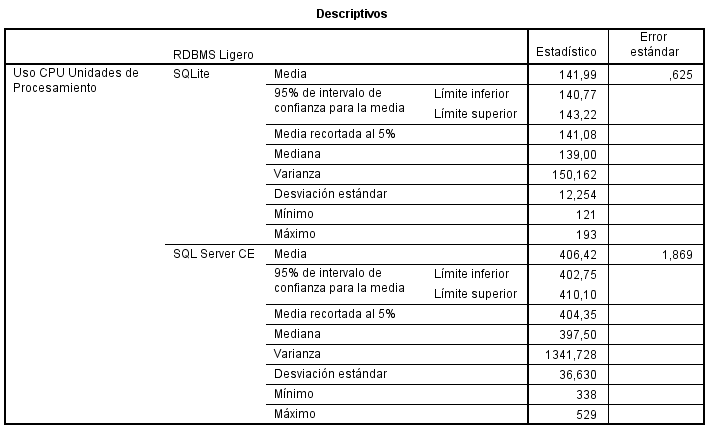
#### **Interpretación de resultados**

Conforme la prueba no paramétrica de hipótesis U de Mann-Whitney, se comprueba que SQL Server Compact presenta diferencias en el uso CPU unidades de tiempos respecto a SQLite, la Tabla 3‑29 indica que tiene un valor más alto en la ponderación respecto al otro, ya que su tiempo promedio en eliminar datos es menor, por lo tanto SQL Server Compact presenta una mayor eficiencia en este caso.

#### **Uso CPU unidades de procesamiento**

Se analiza el uso del recurso CPU unidades de procesamiento que emplea el prototipo Delete tanto para SQLite como para SQL Server CE al momento de su ejecución, en la Tabla 3‑30 indica la estadística descriptiva con los resultados obtenidos con el software estadístico SPSS:

Tabla 3‑30: Estadística descriptiva uso CPU unidades de procesamiento - prototipo Delete



**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

Para SQLite se tiene lo siguiente:

La cantidad de uso en unidades de procesamiento del CPU para este RDBMS ligero es de 141,99 unidades de procesamiento con tendencia a variar ya sea por encima o por debajo de este valor con 0,625 unidades de procesamiento. De entre los valores obtenidos existe menor dispersión de los datos con respecto al promedio de uso de este recurso con 12,254 unidades de procesamiento con valores extremos entre 121 unidades de procesamiento y 193 unidades de procesamiento.

Para SQL Server Compact Edition se tiene:

La cantidad de uso de unidades de procesamiento del CPU para este RDBMS ligero es de 406,42 unidades de procesamiento con una tendencia a variar ya sea por encima o por debajo de este valor con 1,869 unidades de procesamiento. De entre los valores obtenidos existe una dispersión de los datos con respecto al promedio de uso de este recurso con 36,630 con valores extremos entre 338 unidades de procesamiento y 529 unidades de procesamiento.

Para el análisis inferencial de este parámetro se tiene:

**H0**: No existe una diferencia significativa del uso CPU unidades de procesamiento al eliminar datos con SQLite con respecto a SQL Server Compact Edition.

**H1**: Existe una diferencia significativa del uso CPU unidades de procesamiento al eliminar datos con SQLite con respecto a SQL Server Compact.

Con la utilización del software estadístico SPSS la Tabla 3-31 indica la prueba U de Mann Whitney para muestras independientes con los datos obtenidos:

Tabla 3‑31: Estadísticos de contraste uso CPU unidades de procesamiento - prototipo Delete

|  |  |
| --- | --- |
|  | Uso CPU unidades de procesamiento |
| U-Mann Whitney | 147456,000 unidades de procesamiento |
| Significación asintótica (Prueba bilateral) | 0,000 |

FUENTE: Darwin Cedeño y Diego Quinatoa 2015

Mediante esta prueba no paramétrica se rechaza la hipótesis nula H0 y se acepta la alternativa H1, lo cual indica que si existe una diferencia significativa del uso CPU unidades de procesamiento entre las categorías de RDBMS ligeros debido a que el P-valor (significación asintótica) es menor que el alpha (0,000<0.05).

De acuerdo a la ponderación establecida en la Tabla 2‑2: Ponderación de los parámetros, para uso CPU unidades de procesamiento se asigna al RDBMS ligero que satisface las condiciones mencionadas en dicho capítulo con el 25% de los valores y al RDBMS que no satisface se calcula el valor proporcional con una regla de tres inversa con como se indica la Tabla 3-32:

Tabla 3‑32: Ponderación parámetro CPU unidades de tiempo - prototipo Delete

|  |  |  |
| --- | --- | --- |
| PROTIPO | RDBMS | VALOR DE PONDERACION |
| DELETE | SQLite | 25 % |
| SQL Server Compact Edition | 8,73 % |

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

La Figura 3‑10 indica el promedio del uso CPU unidades de procesamiento obtenido con el prototipo Delete para los RDBMS ligeros SQLite y SQL Server Compact, en donde el RDBMS ligero con menor unidades de procesamiento en la ejecución del prototipo es SQLite con una cantidad de trabajo medio de 141,99 unidades de procesamiento a diferencia de SQL Server Compact Edition con 406,42 unidades de procesamiento utilizadas.

Figura 3‑10: Uso CPU unidades de procesamiento - prototipo Delete

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

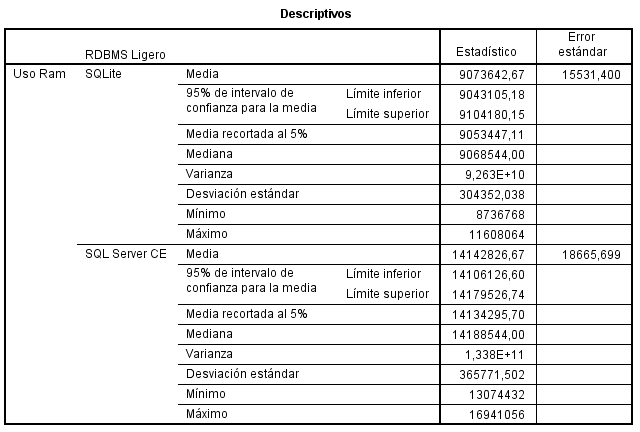
#### **Interpretación de resultados**

Conforme a la prueba no paramétrica de hipótesis U de Mann-Whitney, se comprueba que SQLite presenta diferencias en el uso CPU unidades de procesamiento respecto a SQL Server Compact, la Tabla 3‑32 indica que tiene un valor más alto en la ponderación con respecto al otro, ya que la unidades de procesamiento utilizadas en el ingreso de datos es menor, por lo tanto SQLite presenta una mayor eficiencia en el consumo de este recurso.

#### **Uso RAM**

Se analiza el uso de RAM que emplea el prototipo Delete tanto para SQLite como para SQL Server CE al momento de su ejecución, la Tabla 3‑33 indica la estadística descriptiva con los resultados obtenidos con el software estadístico SPSS:

Tabla 3‑33: Estadística descriptiva uso RAM - prototipo Delete



FUENTE: Darwin Cedeño y Diego Quinatoa 2015

Para SQLite se tiene lo siguiente:

La cantidad del uso RAM para este RDBMS ligero es de 9073642,67 bytes con tendencia a variar ya sea por encima o por debajo de este valor con 15531,400 bytes. De entre los valores obtenidos existe dispersión de los datos con respecto al promedio de uso de este recurso con 304352,038 bytes con valores extremos entre 8736768 bytes y 11608064 bytes.

Para SQL Server Compact Edition se tiene:

La cantidad del uso RAM para este RDBMS ligero es de 14142826,67 bytes con tendencia a variar ya sea por encima o por debajo de este valor con 18665,699 bytes. De entre los valores obtenidos existe una mayor dispersión de los datos con respecto al promedio de uso de este recurso con 365771,502 bytes con valores extremos entre 13074432 bytes y 16941056 bytes.

Para el análisis inferencial de este parámetro se tiene:

**H0**: No existe una diferencia significativa del uso RAM al eliminar datos con SQLite con respecto a SQL Server Compact Edition.

**H1**: Existe una diferencia significativa del uso RAM al eliminar datos con SQLite con respecto a SQL Server Compact.

Con la utilización del software estadístico SPSS, la Tabla 3-34 indica la prueba U de Mann Whitney para muestras independientes con los datos obtenidos:

Tabla 3‑34: Estadísticos de contraste uso RAM - prototipo Delete

|  |  |
| --- | --- |
|  | Uso RAM |
| U-Mann Whitney | 147456,000 bytes |
| Significación asintótica (Prueba bilateral) | 0,000 |

FUENTE: Darwin Cedeño y Diego Quinatoa 2015

Mediante esta prueba no paramétrica se rechaza la hipótesis nula H0 y se acepta la alternativa H1, es decir que si existe una diferencia significativa del uso RAM entre las categorías de RDBMS ligeros debido a que el P-valor (significación asintótica) es menor que el alpha (0,000<0.05).

De acuerdo a la ponderación establecida en la Tabla 2‑2, para uso CPU unidades de procesamiento se asigna al RDBMS ligero que satisface las condiciones mencionadas en dicho capítulo con el 40% de los valores y al RDBMS que no satisface se calcula el valor proporcional con una regla de tres inversa con como se indica en la Tabla 3-35:

Tabla 3‑35: Ponderación parámetro uso RAM - prototipo Delete

|  |  |  |
| --- | --- | --- |
| PROTIPO | RDBMS | VALOR DE PONDERACION |
| DELETE | SQLite | 25 % |
| SQL Server Compact Edition | 16,03 % |

FUENTE: Darwin Cedeño, Diego Quinatoa 2015

La Figura 3‑11 indica el promedio de uso RAM obtenidos con el prototipo Delete para los RDBMS ligeros SQLite y SQL Server Compact Edition, el de menor uso de este recurso en la ejecución de este prototipo es SQLite con una media de 9073642,67 bytes a diferencia de SQL Server Compact Edition con 14142826,67 bytes.

Figura 3‑11: Promedio en bytes del uso del recurso RAM – prototipo Delete

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

#### **Interpretación de resultados**

Conforme a la prueba no paramétrica de hipótesis utilizando U de Mann-Whitney, se comprueba que SQLite presenta diferencias significativas en el uso RAM respecto a SQL Server Compact, además en la Tabla 3‑35 se puede observar que tiene un valor más alto en la ponderación con respecto al otro, ya que de acuerdo a la media del consumo de este recurso en el eliminar datos es menor, por lo tanto SQLite presenta una mayor eficiencia.

#### **Almacenamiento**

Se analiza el uso del recurso Almacenamiento que emplea el prototipo Delete tanto para SQLite como para SQL Server CE al momento de su ejecución, el valor del archivo de almacenamiento no varía tanto para él un RDBMS como para el otro, es un valor constante al realizar las eliminaciones de datos en modo aleatorio, por ende se toma en consideración el de menor tamaño de archivo.

La Tabla 3-36 indica los resultados obtenidos para el uso de este recurso hardware con el tamaño de archivo utilizado:

Tabla 3‑36: Tamaños de archivo Almacenamiento - prototipo Delete

|  |  |
| --- | --- |
| RDBMS | TAMAÑO DE ARCHIVO UTILIZADO |
| SQLite | 209920 bytes |
| SQL Server CE | 327680 bytes |

FUENTE: Darwin Cedeño y Diego Quinatoa 2015

De los valores obtenidos del archivo de almacenamiento al momento de realizar las eliminaciones por cada RDBMS ligero, SQLite es de menor valor con relación a SQL server Compact en el tamaño de archivo utilizado.

De acuerdo a la ponderación establecida en la Tabla 2‑2 para el uso de Almacenamiento, se asigna al RDBMS ligero que satisface las condiciones mencionadas en dicho capítulo con el 10% de los valores y al RDBMS que no satisface se le calcula el valor proporcional con una regla de tres inversa con como se indica en la Tabla 3-37:

Tabla 3‑37: Ponderación parámetro uso Almacenamiento – prototipo Delete

|  |  |  |
| --- | --- | --- |
| PROTIPO | RDBMS | VALOR DE PONDERACION |
| DELETE | SQLite | 10 % |
| SQL Server Compact Edition | 6,4 % |

FUENTE: Darwin Cedeño, Diego Quinatoa 2015

La Figura 3-12 indica el promedio de uso Almacenamiento obtenidos con el prototipo Delete para SQLite y SQL Server Compact Edition:

Figura 3‑12: Promedio del uso Almacenamiento – prototipo Delete

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

#### **Interpretación de resultados**

El RDBMS ligero con menor uso de este recurso en la ejecución de este prototipo es SQLite con un valor de 209920 bytes para el tamaño del archivo a diferencia de SQL Server Compact Edition con 327680 bytes, por lo tanto presenta una mayor eficiencia.

### Eficiencia en Select

#### **Uso CPU unidades de tiempo**

Se analiza la cantidad de uso del recurso CPU unidades de tiempo que emplea el prototipo Select tanto para SQLite como para SQL Server CE al momento su ejecución, la Tabla 3-38 indica la estadística descriptiva con los resultados obtenidos con el software estadístico SPSS:

|  |
| --- |
| Tabla 3‑38: Estadística Descriptiva uso CPU unidades de tiempo - prototipo Select    **FUENTE**: Darwin Cedeño y Diego Quinatoa 2015 |

Para SQLite se tiene lo siguiente:

La cantidad de uso en unidades de tiempo del CPU para este RDBMS es de 1057,13 mili segundos; posee una tendencia a variar ya sea por encima o por debajo de este valor con 4,632 mili segundos. De entre los valores obtenidos existe una mayor dispersión de los datos con respecto al promedio de uso de este recurso con 90,766 mili segundos con valores extremos entre 1035 mili segundos y 2084 mili segundos.

Para SQL Server Compact Edition se tiene:

La cantidad de uso en unidades de procesamiento del CPU para este RDBMS ligero es de 2054,83 mili segundos con una tendencia a variar ya sea por encima o por debajo de este valor con 0,624 mili segundos. De entre los valores obtenidos existe una dispersión de los datos con respecto al promedio de uso de este recurso con 12,231 mili segundos, con valores extremos entre 2020 mili segundos y 2085 mili segundos.

Para el análisis inferencial de este parámetro se tiene:

**H0**: No existe una diferencia significativa del uso CPU unidades de tiempo al listar datos con SQLite con respecto a SQL Server Compact Edition.

**H1**: Existe una diferencia significativa del uso CPU unidades de tiempo al listar datos con SQLite con respecto a SQL Server Compact.

Con la utilización del software estadístico SPSS en la Tabla 3-39 indica la prueba U de Mann Whitney para muestras independientes con los datos obtenidos:

Tabla 3‑39: Estadísticos de contraste uso CPU unidades de tiempo - prototipo Select

|  |  |
| --- | --- |
|  | Uso CPU unidades de tiempo |
| U-Mann Whitney | 146382,500 mili segundos |
| Significación asintótica (Prueba bilateral) | 0,000 |

FUENTE: Darwin Cedeño y Diego Quinatoa 2015

Mediante esta prueba no paramétrica se rechaza la hipótesis nula H0 y se acepta la alternativa H1, indicando que existe una diferencia significativa de uso del CPU unidades tiempo entre las categorías de RDBMS ligeros debido a que el P-valor (Significación asintótica) es menor que el alpha (0,000<0.05).

De acuerdo a la ponderación establecida en la Tabla 2‑2 para uso CPU unidades de tiempo, se asigna al RDBMS ligero que satisface las condiciones mencionadas en dicho capítulo con el 25% de los valores y al RDBMS que no satisface se calcula el valor proporcional con una regla de tres inversa con como se indica en la Tabla 3-40:

Tabla 3‑40: Ponderación parámetro uso CPU unidades de tiempo - prototipo Select

|  |  |  |
| --- | --- | --- |
| PROTIPO | RDBMS | VALOR DE PONDERACION |
| SELECT | SQLite | 25 % |
| SQL Server Compact Edition | 12,86 % |

FUENTE: Darwin Cedeño, Diego Quinatoa 2015

La Figura 3‑13 indica el promedio de las unidades de tiempo en uso del CPU obtenidos con el prototipo Select para SQLite y SQL Server Compact Edition, en donde el RDBMS ligero con menores unidades de tiempo de uso CPU para la ejecución del prototipo es SQLite con un tiempo medio de 1057,13ms a diferencia de SQL Server Compact Edition con 2054,83ms.

Figura 3‑13: Promedio de uso CPU unidades de tiempo (ms) - prototipo Select

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

#### **Interpretación de resultados**

Conforme la prueba no paramétrica de hipótesis utilizando U de Mann-Whitney, se comprueba que SQLite presenta diferencias en el uso CPU unidades de tiempos respecto a SQL Server Compact, además en la Tabla 3‑40 indica que tiene un valor más alto en la ponderación con respecto al otro, ya que su tiempo promedio al listar datos es menor, por lo tanto SQLite presenta una mayor eficiencia.

#### **Uso CPU unidades de procesamiento**

Se analiza la cantidad de uso del recurso CPU unidades de procesamiento que emplea el prototipo Select tanto para SQLite como para SQL Server CE al momento de su ejecución, la Tabla 3-41se indica la estadística descriptiva con los resultados obtenidos con el software estadístico SPSS:

Tabla 3‑41: Estadística Descriptiva uso CPU unidades de procesamiento - prototipo Select

|  |
| --- |
| **FUENTE**: Darwin Cedeño y Diego Quinatoa 2015 |

Para SQLite se tiene lo siguiente:

La cantidad de uso en unidades de procesamiento del CPU para este RDBMS ligero es de 74,84 unidades de procesamiento con tendencia a variar ya sea por encima o por debajo de este valor con 0,233 unidades de procesamiento. De entre los valores obtenidos existe menor dispersión de los datos con respecto al promedio de uso de este recurso con 4,557 unidades de procesamiento con valores extremos entre 66 unidades de procesamiento y 111 unidades de procesamiento.

Para SQL Server Compact Edition se tiene:

La cantidad de uso en unidades de procesamiento del CPU para este RDBMS ligero es de 112,71 unidades de procesamiento con tendencia a variar ya sea por encima o por debajo de este valor con 0,299 unidades de procesamiento. De entre los valores obtenidos existe una dispersión de los datos con respecto al promedio de uso de este recurso con 5,850 unidades de procesamiento con valores extremos entre 99 unidades de procesamiento y 136 unidades de procesamiento.

Para el análisis inferencial de este parámetro se tiene:

**H0**: No existe una diferencia significativa del uso CPU unidades de procesamiento al listar datos con SQLite con respecto a SQL Server Compact Edition.

**H1**: Existe una diferencia significativa del uso CPU unidades de procesamiento al listar datos de SQLite con respecto a SQL Server Compact.

Con la utilización del software estadístico SPSS la Tabla 3-42 indica la prueba U de Mann Whitney para muestras independientes con los datos obtenidos:

Tabla 3‑42: Estadísticos de contraste uso CPU unidades de procesamiento – prototipo Select

|  |  |
| --- | --- |
|  | Uso CPU unidades de procesamiento |
| U-Mann Whitney | 147231,500 unidades de procesamiento |
| Significación asintótica (Prueba bilateral) | 0,000 |

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

Mediante esta prueba no paramétrica se rechaza la hipótesis nula H0 y se acepta la alternativa H1, es decir que si existe una diferencia significativa de uso CPU unidades procesamiento entre las categorías de RDBMS ligeros debido a que el P-valor (significación asintótica) es menor que el alpha (0,000<0.05).

De acuerdo a la ponderación establecida en la Tabla 2‑2 para uso CPU unidades de procesamiento, se asigna al RDBMS ligero que satisface las condiciones mencionadas en dicho capítulo con el 25% de los valores y al RDBMS que no satisface se calcula el valor proporcional con una regla de tres inversa como indica en la tabla 3-43:

Tabla 3‑43: Ponderación parámetro uso CPU unidades de procesamiento - prototipo Select

|  |  |  |
| --- | --- | --- |
| PROTIPO | RDBMS | VALOR DE PONDERACION |
| SELECT | SQLite | 25 % |
| SQL Server Compact Edition | 16,6 % |

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

La Figura 3‑14 indica el promedio de uso CPU unidades de procesamiento obtenidos con el prototipo Select para SQLite y SQL Server Compact, en donde el RDBMS ligero con menores unidades de procesamiento en la ejecución del prototipo es SQLite con una cantidad de trabajo medio de 74,84 unidades de procesamiento a diferencia de SQL Server Compact Edition con 112,71 unidades de procesamiento utilizadas.

Figura 3‑14: Uso CPU unidades de procesamiento – prototipo Select

FUENTE: Darwin Cedeño y Diego Quinatoa 2015

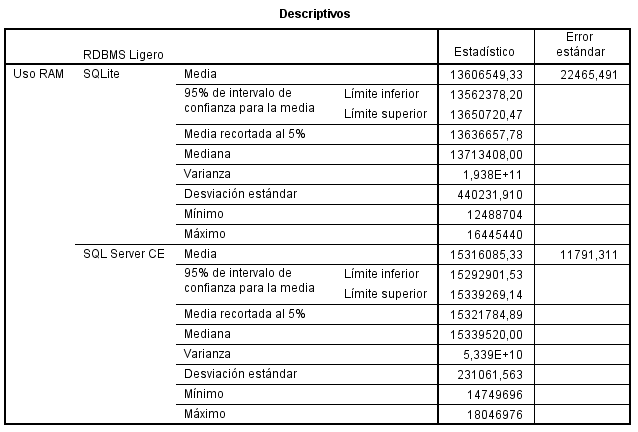
#### **Interpretación de resultados**

Conforme a la prueba no paramétrica de hipótesis U de Mann-Whitney, se comprueba que SQLite presenta diferencias en el uso CPU unidades de procesamiento respecto a SQL Server Compact, la Tabla 3‑43 indica que tiene un valor más alto en la ponderación con respecto al otro, ya que las unidades de procesamiento utilizadas al listar datos es menor, por lo tanto SQLite presenta una mayor eficiencia.

#### **Uso de RAM**

Se analiza la cantidad de uso del recurso RAM que emplea la ejecución del prototipo Select tanto para SQLite como para SQL Server CE, la Tabla 3‑44 indica la estadística descriptiva con los resultados obtenidos para el uso de RAM con el software estadístico SPSS:

Tabla 3‑44: Estadística Descriptiva uso RAM – prototipo Select



**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

Para SQLite se tiene lo siguiente:

El pico de uso máximo RAM para este RDBMS ligero es de 13606549,33 bytes con tendencia a variar ya sea por encima o por debajo de este valor con 22465,491 bytes. De entre los valores obtenidos existe dispersión de los datos con respecto al promedio de uso de este recurso con 440231,910 bytes con valores extremos entre 12488704 bytes y 16445440 bytes.

Para SQL Server Compact Edition se tiene:

El pico de uso máximo RAM para este RDBMS ligero es de 15316085,33 bytes con tendencia a variar ya sea por encima o por debajo de este valor con 11791,311 bytes. De entre los valores obtenidos existe una menor dispersión de los datos con respecto al promedio de uso de este recurso con 231061,563 bytes con valores extremos entre 14749696 bytes y 18046976 bytes

Para el análisis inferencial de este parámetro se tiene:

**H0**: No existe una diferencia significativa del uso RAM al listar datos con SQLite con respecto a SQL Server Compact Edition.

**H1**: Existe una diferencia significativa del uso RAM al listar datos con SQLite con respecto a SQL Server Compact.

Con la utilización del software estadístico SPSS se hace el análisis de los datos, la tabla 3-45 indica la prueba U de Mann Whitney para muestras independientes con los datos obtenidos:

Tabla 3‑45: Estadísticos de contraste uso RAM – prototipo Select

|  |  |
| --- | --- |
|  | Uso RAM |
| U-Mann Whitney | 146310,000 bytes |
| Significación asintótica (Prueba bilateral) | 0,000 |

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

Mediante esta prueba no paramétrica se rechaza la hipótesis nula H0 y se acepta la alternativa H1, es decir que existe un diferencia significativa de uso RAM entre las categorías de RDBMS ligeros debido a que el P-valor (significación asintótica) es menor que el alpha (0,000<0.05). De acuerdo a la ponderación establecida en la Tabla 2‑2 para uso RAM, se asigna al RDBMS ligero que satisface las condiciones mencionadas en dicho capítulo con el 40% de los valores y al RDBMS que no satisface se calcula el valor proporcional con una regla de tres inversa con como se indica la Tabla 3-46:

Tabla 3‑46: Ponderación parámetro uso RAM – prototipo Select

|  |  |  |
| --- | --- | --- |
| PROTIPO | RDBMS | VALOR DE PONDERACION |
| SELECT | SQLite | 40 % |
| SQL Server Compact Edition | 35,54 % |

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

La Figura 3-15 indica el promedio de uso RAM obtenidos con el prototipo SELECT, en donde el RDBMS ligero con menor uso de este recurso en la ejecución de este prototipo es SQLite con una media de 13606549,33 bytes a diferencia de SQL Server Compact Edition con 15316085,33 bytes.

Figura 3‑15: Promedio en bytes del uso del recurso RAM – prototipo Select

FUENTE: Darwin Cedeño y Diego Quinatoa 2015

#### **Interpretación de resultados**

Conforme a la prueba no paramétrica de hipótesis U de Mann-Whitney, se comprueba que SQLite presenta diferencias significativas en el uso RAM respecto a SQL Server Compact, la Tabla 3‑46 indica que tiene un valor más alto en la ponderación con respecto al otro, ya que de acuerdo al valor de la media del consumo de este recurso al listar datos es menor, por lo tanto SQLite presenta una mayor eficiencia.

#### **Almacenamiento**

Se analiza el uso del recurso Almacenamiento que emplea el prototipo Select tanto para SQLite como para SQL Server CE al momento de su ejecución, el valor del archivo de almacenamiento no varía tanto para él un RDBMS como para el otro, es un valor constante al realizar las eliminaciones de datos en modo aleatorio, por ende se toma en consideración el de menor tamaño de archivo. La Tabla 3-47 indica los resultados obtenidos para el uso de este recurso hardware con el tamaño de archivo utilizado:

Tabla 3‑47: Tamaños de archivo Almacenamiento – prototipo Select

|  |  |
| --- | --- |
| RDBMS | TAMAÑO DE ARCHIVO UTILIZADO |
| SQLite | 209920 bytes |
| SQL Server CE | 327680 bytes |

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

De los valores obtenidos del archivo de almacenamiento al momento de listar datos por cada RDBMS ligero SQLite es de menor valor con relación a SQL server Compact en el tamaño de archivo utilizado.

De acuerdo a la ponderación establecida en la Tabla 2‑2 para el uso de Almacenamiento, se asigna al RDBMS ligero que satisface las condiciones mencionadas en dicho capítulo con el 10% de los valores y al RDBMS que no satisface se le calcula el valor proporcional con una regla de tres inversa con como se indica en la tabla 3-48:

Tabla 3‑48: Ponderación parámetro uso Almacenamiento – prototipo Select

|  |  |  |
| --- | --- | --- |
| PROTIPO | RDBMS | VALOR DE PONDERACION |
| INSERTAR | SQLite | 10 % |
| SQL Server Compact Edition | 6,4 % |

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

La Figura 3-16 indica el promedio de uso Almacenamiento obtenido con el prototipo Select:

Figura 3‑16: Promedio del uso Almacenamiento – prototipo Select

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

#### **Interpretación de resultados**

El RDBMS ligero con menor uso de este recurso en la ejecución de este prototipo es SQLite con un valor de 209920 bytes para el tamaño del archivo a diferencia de SQL Server Compact Edition con 327680 bytes, por lo tanto presenta una mayor eficiencia.

En la tabla 3-49 se puede ver la estadística descriptiva para el RDBMS SQLite

Tabla 3‑49: Resultado de las medias de uso de los recursos hardware obtenidas con la estadística descriptiva para el RDBMS SQLite

|  |  |  |
| --- | --- | --- |
|  | Recurso Hardware | Medias |
| Eficiencia en Insert | Uso CPU unidades de tiempo | 18858,36 ms |
| Uso CPU unidades de procesamiento | 157,99 unidades de procesamiento |
| Uso RAM | 9384565,33 bytes |
| Almacenamiento | 209920 bytes |
| Eficiencia en Update | Uso CPU unidades de tiempo | 18489,69 ms |
| Uso CPU unidades de procesamiento | 161,55 unidades de procesamiento |
| Uso RAM | 1134549,33 bytes |
| Almacenamiento | 209920 bytes |
| Eficiencia en Delete | Uso CPU unidades de tiempo | 17522,34 ms |
| Uso CPU unidades de procesamiento | 141,99 unidades de procesamiento |
| Uso RAM | 9073642,67 bytes |
| Almacenamiento | 209920 bytes |
| Eficiencia en Select | Uso CPU unidades de tiempo | 1057,13 ms |
| Uso CPU unidades de procesamiento | 74,84 unidades de procesamiento |
| Uso RAM | 13606549,33 bytes |
| Almacenamiento | 209920 bytes |

En la tabla 3-50 se puede ver la estadística descriptiva para el RDBMS SQL Server Compact

Tabla 3‑50: Resultado de las medias de uso de los recursos hardware obtenidas con la estadística descriptiva para el RDBMS SQL Server Compact

|  |  |  |
| --- | --- | --- |
|  | Recurso Hardware | Medias |
| Eficiencia en Insert | Uso CPU unidades de tiempo | 22802,33 ms |
| Uso CPU unidades de procesamiento | 827,89 unidades de procesamiento |
| Uso RAM | 12952938,67 bytes |
| Almacenamiento | 327680 bytes |
| Eficiencia en Update | Uso CPU unidades de tiempo | 19916,47 ms |
| Uso CPU unidades de procesamiento | 798,82 unidades de procesamiento |
| Uso RAM | 13988384,00 bytes |
| Almacenamiento | 327680 bytes |
| Eficiencia en Delete | Uso CPU unidades de tiempo | 17085,20 ms |
| Uso CPU unidades de procesamiento | 406,42 unidades de procesamiento |
| Uso RAM | 14142826,67 bytes |
| Almacenamiento | 327680 bytes |
| Eficiencia en Select | Uso CPU unidades de tiempo | 2054,83 ms |
| Uso CPU unidades de procesamiento | 112,71 unidades de procesamiento |
| Uso RAM | 15316085,33 bytes |
| Almacenamiento | 327680 bytes |

## Resultados generales

La Tabla 3-51 indica los resultados generales de la investigación con los promedios en eficiencia obtenidos a través de las pruebas realizadas por cada RDBMS ligero:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| RDBMS Ligero | Eficiencia en Insert | Eficiencia en Update | Eficiencia en Delete | Eficiencia en Select | PROMEDIO TOTAL |
| SQLite | 100% | 100% | 84,37% | 100% | 96.09% |
| SQL Server Compact | 49,76% | 66,5% | 56,16% | 71,4% | 60.95% |

Tabla 3‑51: Resultados generales

FUENTE: Darwin Cedeño, Diego Quinatoa 2015

En la figura 3-17 se representa gráficamente estos promedios obtenidos:

Figura 3‑17: Promedios de eficiencia por cada RDBMS ligero.

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

**Interpretación de resultados**

Observando los resultados de la Figura 3‑17 existe una diferencia significativa entre los RDBMS ligeros SQLite con un promedio del 96,09% y SQL Server CE con 60,95%, lo que implica que analizando estos promedios SQLite presenta mejores resultados en cuanto a eficiencia ya que hace uso de mejor de mejor manera los recursos hardware.

## Demostración de hipótesis

En esta investigación, la hipótesis que se planteo es la siguiente:

“El RDBMS ligero SQLite es más eficiente en el consumo de recursos hardware que SQL Server Compact Edition para el desarrollo del módulo móvil de gestión de pedidos del sistema de facturación del hotel Los Lirios”.

Con relación a esto se define:

: “Los RDBMS ligeros SQLite y SQL Server Compact son igual de eficientes en el consumo de recursos hardware”.

Como hipótesis alternativa se tiene:

: “Los RDBMS ligeros SQLite y SQL Server Compact son diferentes en la eficiencia del consumo de recursos hardware”.

Según el análisis inferencial y los resultados que indica la tabla 3-51, los RDBMS ligeros comparados en eficiencia son diferentes, rechazando la hipótesis nula y aceptando la alternativa, lo que implica que SQLite es más eficiente en el consumo de recurso hardware concluyendo así con la aceptación de la hipótesis planteada de la investigación.

# CAPÍTULO IV

# SISTEMA DE FACTURACIÓN Y ADMINISTRACIÓN DEL HOTEL “LOS LIRIOS” (Lirios Fadmin)

El siguiente capítulo describe el proceso de desarrollo del Sistema de Facturación y Administración del hotel “Los Lirios” con un módulo móvil de gestión de pedidos para Windows Phone.

## Hotel “Los Lirios” panorama general

El hotel “Los Lirios” es una microempresa ubicada en la ciudad del Tena que ofrece hospedaje y servicios varios a sus clientes, actualmente no cuenta con un sistema informático para la facturación y administración de dichos servicios. La dinámica actual con la que viene trabajando el hotel se presentan necesidades día a día tales como:

* La facturación que se realiza a mano lo cual provoca la inexactitud de los registros.
* La pérdida de tiempo que genera al cliente molestias cuando solicita una factura y este no tiene conocimiento de los valores a pagar con exactitud, razón de su inconformidad con el servicio de facturación.
* La veracidad de dicha información al momento de registrar los pedidos que hizo el cliente, datos de gran importancia para el usuario en cuanto a los pedidos realizados.
* Incomodidad cuando un huésped solicita servicios a la habitación ya que la persona debe acercarse a la recepción para realizar algún pedido.

Luego de definir la problemática mencionada los autores de esta investigación aceptan el proyecto de realización del sistema de facturación y administración LiriosFadmin con un módulo móvil LiriosMobile para la gestión de pedidos.

Para el desarrollo del sistema se utilizó la metodología SCRUM dado que es una metodología de software ágil que permite entregas de partes de un producto con una flexibilidad y adaptación a los cambios que se producen durante la implementación del proyecto por parte del cliente.

## Roles SCRUM

Esta metodología se basa en el trabajo en equipo para lo cual se definen roles que se encargan de ayudar a que el proyecto culmine de manera satisfactoria. Con el escenario planteado para la realización del sistema de facturación y administración se indican los siguientes roles Scrum:

* **Product Owner (propietario del producto)**

La señora Rosa Cando Gerente del hotel “Los Lirios”

* **Team (Equipo)**

El equipo de desarrollo compuesto por Darwin Cedeño y Diego Quinatoa.

* **Scrum Master**

Ingeniero Jorge Menéndez.

## Planificación

En ésta fase del proyecto se realizó la planificación de requerimientos tras estudiar, evaluar y mantener reuniones constantes con el Product Owner (propietario del producto), estos son representados como historias de usuario en donde estas historias fueron eliminadas o modificadas a medida que se suscitaban cambios en los requisitos del cliente o se tenía una idea más clara del sistema.

### Product Backlog

El Product Backlog listará de manera priorizada todas las funcionalidades del sistema LiriosFadmin dando un orden a las historias de usuario teniendo en consideración primero las de mayor prioridad y sucesivamente las de menor prioridad, esta prioridad se estableció con la ayuda del product owner como: 1 = Muy Baja, 2= Baja, 3= Normal, 4=Alto, 5=Muy Alto.

El identificador de cada historia de usuario sigue las siguientes convenciones:

* Historia técnica con identificación HT <número>
* Historia de usuario de LiriosDesktop con identificación HU <número>
* Historia de usuario de LiriosMobile con identificación HUM <número>

Los puntos de estimación de cada historia de usuario se estableció utilizando el método *T-Shirt Sizing* con las tallas comunes de camisetas donde: Small (S) = 1 punto, Medium (M) = 3 puntos, Large (L) = 4 puntos y Extra Large (XL) = 5 puntos. La tabla 4-1 indica el product backlog de todo el proyecto:

Tabla 4‑1: Product Backlog LiriosFadmin

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Nombre de la Historia | Descripción | Prioridad | Puntos |
| HT1 | **Diseño de la Base de Datos LiriosFadminDB** | Como administrador deseo diseñar la base de datos para almacenar la información. | 5 | 5 |
| HT2 | **Diseño de la arquitectura del sistema** | Como administrador deseo diseñar la arquitectura del sistema para tener mejor visión de la solución. | 5 | 3 |
| HT3 | **Instalación de herramientas** | Como administrador instalar las herramientas necesarias para el desarrollo del sistema. | 5 | 1 |
| HT4 | **Diseño de interfaces de usuario** | Como administrador realizar el diseño de las interfaces de usuario | 5 | 1 |
| HT5 | **Codificación** | Como administrador seleccionar un estándar de codificación. | 5 | 4 |
| HU1 | **Inicio de sesión** | Como administrador deseo que exista un inicio de sesión para validar y controlar el acceso autorizado al sistema. | 5 | 4 |
| HU2 | **Registro de Usuario** | Como administrador deseo registrar un usuario para controlar el acceso a los distintos módulos del sistema. | 5 | 3 |
| HU3 | **Actualizar un Usuario** | Como administrador deseo actualizar los datos del usuario para tener información actual. | 5 | 3 |
| HU4 | **Registrar las características de habitación** | Como administrador o gerente deseo registrar las características de un tipo de habitación para saber qué clase de servicio tiene el hotel. | 5 | 3 |
| HU5 | **Actualizar una característica de habitación** | Como administrador o gerente deseo actualizar las características de un tipo de habitación para tener una información actual. | 5 | 3 |
| HU6 | **Eliminar una característica de habitación** | Como administrador o gerente deseo eliminar las características de un tipo de habitación para tener una información actualizada. | 5 | 3 |
| HU7 | **Registrar un tipo de habitación** | Como administrador o gerente deseo registrar un tipo de habitación para saber qué clase de servicio tiene el hotel. | 5 | 4 |
| HU8 | **Actualizar un tipo de habitación** | Como administrador o gerente deseo actualizar un tipo de habitación para tener una información actual. | 5 | 4 |
| HU9 | **Eliminar un tipo de habitación** | Como administrador o gerente deseo eliminar un tipo de habitación para tener una información actualizada. | 5 | 3 |
| HU10 | **Registrar una habitación** | Como administrador o gerente deseo registrar una habitación para saber qué servicios tiene el hotel. | 4 | 5 |
| HU11 | **Registrar un producto** | Como administrador o gerente deseo registrar un producto para la venta a los usuarios. | 4 | 4 |
| HU12 | **Actualizar un producto** | Como administrador o gerente deseo actualizar un producto para tener información actualizada. | 4 | 5 |
| HU13 | **Registro de un cliente** | Como recepcionista deseo registrar cliente para poder ofrecer servicios del hotel. | 4 | 5 |
| HU14 | **Actualizar un cliente** | Como recepcionista deseo actualizar un cliente para obtener información actual | 4 | 5 |
| HU15 | **Crear una cuenta de servicio** | Como recepcionista deseo crear una cuenta de servicios para registrar distintos servicios del cliente. | 3 | 4 |
| HU16 | **Agregar un servicio de hospedaje** | Como recepcionista deseo agregar un servicio de hospedaje para asociar a una cuenta de servicio de un cliente. | 3 | 4 |
| HU17 | **Agregar una venta de un producto** | Como recepcionista deseo agregar una venta de un producto para asociar a una cuenta de servicio de un cliente | 3 | 4 |
| HU18 | **Finalizar un servicio de hospedaje** | Como recepcionista deseo finalizar un servicio de hospedaje para cambiar el estado de una habitación de ocupado a mantenimiento y posterior uso nuevamente. | 3 | 5 |
| HU19 | **Cancelación de un servicio de hospedaje** | Como gerente deseo cancelar un servicio de hospedaje para no considerar como cobro por posibles errores de ingreso de datos. | 3 | 5 |
| HU20 | **Aplicar descuento del 100%** | Como gerente deseo aplicar un descuento del 100% para ofrecer hospedaje de cortesía a determinado cliente. | 3 | 5 |
| HU21 | **Agregar abonos** | Como recepcionista deseo agregar abonos para disminuir el valor a pagar en la cuenta final. | 3 | 3 |
| HU22 | **Finalizar cuenta de servicio** | Como Gerente, recepcionista, o administrador deseo la cuenta de servicio del cliente para proceder con el cobro de valores y su posterior facturación. | 3 | 5 |
| HU23 | **Visualizar detalle de la cuenta de servicios** | Como Gerente, recepcionista o administrador deseo visualizar el detalle de la de servicios del cliente para indicar los valores a pagar. | 3 | 5 |
| HUM1 | **Inicio de sesión Cliente** | Como cliente deseo iniciar sesión en la aplicación móvil para poder acceder a los servicios del hotel. | 2 | 4 |
| HUM2 | **Realizar pedido a la habitación** | Como cliente deseo realizar un pedido a la habitación a través de la aplicación móvil para cubrir las necesidades y deseos del cliente. | 2 | 4 |
| HUM3 | **Consulta de pedidos realizados** | Como cliente deseo listar los pedidos realizados para visualizar como cliente lo que he solicitado. | 2 | 5 |
| HUM4 | **Detalle de la cuenta de servicios** | Como cliente deseo visualizar el detalle de la cuenta de servicios para tener en cuenta lo que se debe pagar. | 2 | 3 |
| HUM5 | **Inicio de sesión camarero** | Como camarero deseo iniciar sesión en la aplicación móvil para realizar actividades en la habitación | 2 | 3 |
| HUM6 | **Entrega de un producto** | Como camarero deseo registrar la entrega de un producto para confirmar el pedido. | 2 | 4 |
| HU24 | **Visualizar listado de habitaciones** | Como administrador, recepcionista o gerente deseo visualizar el listado de habitaciones en mantenimiento para enviar a una camarera que realiza su respectivo trabajo. | 1 | 3 |
| HU25 | **Actualizar el estado de una habitación** | Como administrador, recepcionista o gerente deseo actualizar el estado de una habitación a disponible para incluirla de nuevo en los servicios del hotel. | 1 | 3 |
| HU26 | **Notificaciones recepcionista** | Como recepcionista deseo visualizar notificaciones referente a las habitaciones en mantenimiento, como pedidos a la habitación. | 1 | 5 |
| HT6 | **Manual de usuario** | Como administrador elaborar el manual de usuario | 1 | 3 |

FUENTE: Darwin Cedeño, Diego Quinatoa 2015

El Product Backlog contiene en total 38 historias de usuario, donde 6 corresponden a las historias técnicas (diseño de base de datos, arquitectura del sistema, instalación de herramientas, interfaces de usuario, codificación y manual de usuario), 26 historias de usuario que definen los requerimientos específicos del sistema LiriosFadmin con un total de 141 puntos estimados en todas las historias.

### Planificación de entregables.

En este fase del proyecto se realiza la planificación de los entregables de las historias de usuario e historias técnicas descritas en el Product Backlog de la Tabla 4‑1.

La Tabla 4-2 indica en una lista 4 entregables que se han establecido de todas las historias de usuario e historias técnicas asignadas a cada uno de los Sprints.

Tabla 4‑2: Sprint Backlog LiriosFadmin.

|  |  |
| --- | --- |
| SPRINT 1  7/12/2015 – 18/12/2015 | |
| HT1 | Diseño de la Base de Datos LiriosFadminDB |
| HT2 | Diseño de la arquitectura del sistema |
| HT3 | Instalación de herramientas |
| HT4 | Diseño de interfaces de usuario |
| HT5 | Codificación |
| HU1 | Inicio de sesión |
| HU2 | Registro de Usuario |
| HU3 | Actualizar un Usuario |
| SPRINT 2  11/01/2016 – 22/01/2016 | |
| HU7 | Registrar un tipo de habitación |
| HU8 | Actualizar un tipo de habitación |
| HU9 | Eliminar un tipo de habitación |
| HU10 | Registrar una habitación |
| HU11 | Registrar un producto |
| HU12 | Actualizar un producto |
| HU13 | Registro de un cliente |
| HU14 | Actualizar un cliente |
| SPRINT 3  25/01/2016 – 05/02/2016 | |
| HU15 | Crear una cuenta de servicio |
| HU16 | Agregar un servicio de hospedaje |
| HU17 | Agregar una venta de un producto |
| HU18 | Finalizar un servicio de hospedaje |
| HU19 | Cancelación de un servicio de hospedaje |
| HU20 | Aplicar descuento del 100% |
| HU21 | Agregar abonos |
| HU22 | Finalizar cuenta de servicio |
| HU23 | Visualizar detalle de la cuenta de servicios |
| SPRINT 4  15/02/2016 – 26/02/2016 | |
| HUM1 | Inicio de sesión Cliente |
| HUM2 | Realizar pedido a la habitación |
| HUM3 | Consulta de pedidos realizados |
| HUM4 | Detalle de la cuenta de servicios |
| HUM5 | Inicio de sesión camarero |
| HUM6 | Entrega de un producto |
| HU24 | Visualizar listado de habitaciones |
| HU25 | Actualizar el estado de una habitación |
| HU26 | Notificaciones recepcionista |
| HT6 | Manual de usuario |

FUENTE: Darwin Cedeño, Diego Quinatoa 2015

El Sprint 1, contiene 5 historias técnicas las cuales corresponden al diseño de la solución del sistema de facturación y administración del hotel “Los Lirios” y tres historias de usuario como introducción al desarrollo de las funcionalidades de LiriosWebService. El Sprint 2 contiene 8 historias de usuario escogidas para este Sprint, en este se continúa con el desarrollo de la funciones del servicio web LiriosWebService.

El Sprint 3 contiene 9 historias de usuario escogidas para este Sprint, en este se continúa con el desarrollo de las funciones del servicio web LiriosWebService.

Y por último el Sprint 4 con 10 de historias de usuario escogidas para este Sprint, divididas en 1 historias técnica para los manuales, 6 historias de usuario que representan el módulo móvil del sistema de facturación para la gestión de pedidos y 3 historias más que representan la continuación del desarrollo de las funciones Lirios Desktop. Se concluye que con esta planificación se consiguió cumplir con las expectativas de desarrollo de cada entregable cumpliendo con los tiempos establecidos.

## Desarrollo

Esta fase del proyecto comprende la ejecución de cada uno de los Sprint obtenidos en el Product Backlog del sistema. Durante 4 reuniones con el product owner se plantearon y definieron las historias de usuario con el formato que indica la tabla 4-3, con un identificador, un nombre, prioridad, puntos de estimación, un tipo de usuario para la historia, un responsable, tareas de ingeniería y la descripción.

Las tarjetas de historia de usuario del proyecto se las puede individuar en el anexo B.

Tabla 4‑3: Formato tarjetas de historia de usuario.

|  |  |
| --- | --- |
| HISTORIA DE USUARIO | |
| Id | HT1 |
| Nombre | Diseño de la base de datos LiriosFadminDB |
| Prioridad | 5 |
| Puntos de estimación | 5 |
| Descripción | Como administrador deseo diseñar la base de datos para almacenar la información. |
| Tareas de ingeniería | - Generar el diagrama de la base de datos en papel.  -Generar el Script de la base de datos. |
| Responsable | Equipo |
| Tipo de Usuario | Administrador |

FUENTE: Darwin Cedeño, Diego Quinatoa 2015

Con la obtención del product backlog se procedió a la creación de los Sprints de 2 semanas cada uno con 5 días laborales a un total de 8 horas por día. Para determinar las historias que se deben desarrollar se realizaron reuniones previas a cada sprint considerando la prioridad del product owner y la capacidad y experiencia del equipo. En total fueron 4 Sprints los cuales se detallan en el anexo B.

Al finalizar cada sprint, al product owner se mostró lo desarrollado, se ejecutaron pruebas finales para realizar validaciones y posteriormente obtener la aceptación por parte del mismo esto expresado en tarjetas de pruebas de aceptación detalladas en el anexo C. En la tabla 4-4 se indica un modelo de prueba de aceptación.

Tabla 4‑4: Modelo de prueba de aceptación.

|  |  |
| --- | --- |
| PRUEBA DE ACEPTACIÓN | |
| Id | PA1 |
| Historia a la que pertenece | HT1 |
| Descripción | Diseño de la Base de Datos LiriosFadminDB |
| Precondiciones | ninguna |
| Ejecución | 1.- Diseñar tablas.  2.- Generar el Script de la base de datos con tablas, funciones y data preliminar. |
| Resultado | Base de datos generada. |
| Responsable | Darwin Cedeño |
| Evaluación | Exitosa. |

FUENTE: Darwin Cedeño, Diego Quinatoa 2015

**Sprint 1**

El primer Sprint comprende la definición de la solución para el proyecto en donde se procede con el desarrollo de las funcionalidades del sistema LiriosFadmin. Contiene 5 historias técnicas y 6 historia de usuario, dichas historias para este sprint se indican en la Tabla 4-5.

Tabla 4‑5: Sprint Backlog–Sprint 1- Diseño de la solución, LiriosFadmin

|  |  |
| --- | --- |
| SPRINT 1 | |
| HT1 | Diseño de la Base de Datos LiriosFadminDB |
| HT2 | Diseño de la arquitectura del sistema |
| HT3 | Instalación de herramientas |
| HT4 | Diseño de interfaces de usuario |
| HT5 | Codificación |
| HU1 | Inicio de sesión |
| HU2 | Registro de Usuario |
| HU3 | Actualizar un Usuario |
| HU4 | Registrar las características de habitación |
| HU5 | Actualizar una característica de habitación |
| HU6 | Eliminar una característica de habitación |

FUENTE: Darwin Cedeño, Diego Quinatoa 2015.

**Diseño de la Arquitectura del sistema**

Para el diseño de la arquitectura del sistema la cual se detalla en Figura 4‑1 se indica el sistema en general LiriosFadmin con componentes necesarios para la estructura tales como:

* Una equipo con Windows 8.1 instalado que hará de servidor de bases de datos (LiriosDB) con SQL Server 2012 y servidor web (Internet Information Service 8.5) que manejará todo lo relacionado al servicio web (LiriosWebService) el cual gestionará las peticiones que hacen los clientes a través de la red local del hotel. El servicio web se conecta al servidor de bases de datos para obtener o manipular la información.
* Los dispositivos clientes : Un computador en la recepción del hotel con Windows 8.1 o superior para la aplicación de escritorio (Lirios Desktop) y los dispositivos móviles con Windows Phone 8.1 para instalar la aplicación móvil (Lirios Mobile) .

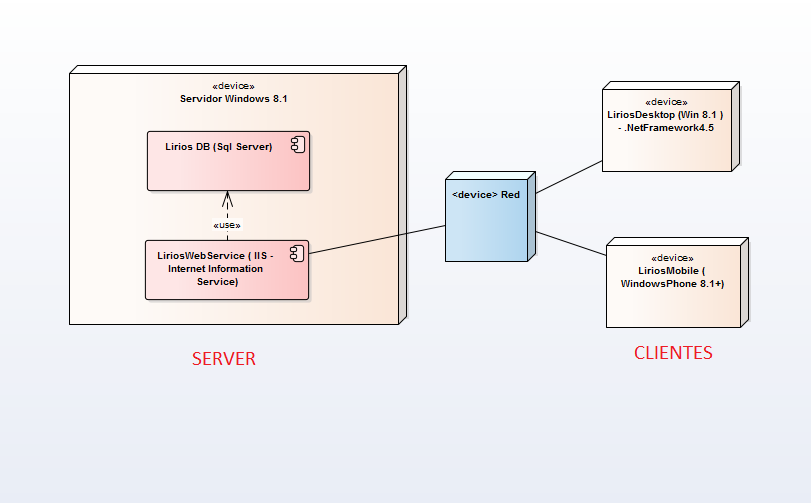


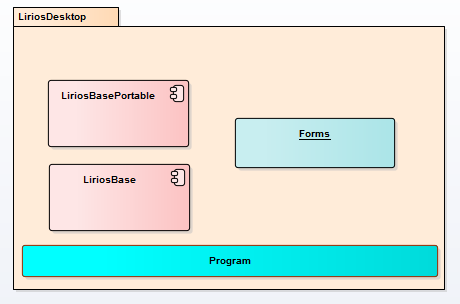
Figura 4‑1: Arquitectura LiriosFadmin

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

**Arquitectura LiriosDesktop**

La arquitectura de LiriosDesktop se puede observar en la Figura 4-2, la misma que posee los siguientes elementos:

* **LiriosBasePortable**: Librería que contiene las clases base de la lógica de negocio de la aplicación compatible con aplicaciones web, escritorio y móvil.
* **LiriosBase**: Contiene las clases base propias de la aplicación compatible solo con aplicaciones web y escritorio.
* **Forms**: Formularios que constituyen la interfaz gráfica de la aplicación
* **Program**: Clase principal encargada del inicio de la ejecución de un programa.



**Figura 4‑2**: Arquitectura LiriosDesktop

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

**Arquitectura LiriosMobile**

La arquitectura de LiriosMobile se puede observar en la figura 4-3, la misma que posee los siguientes elementos:

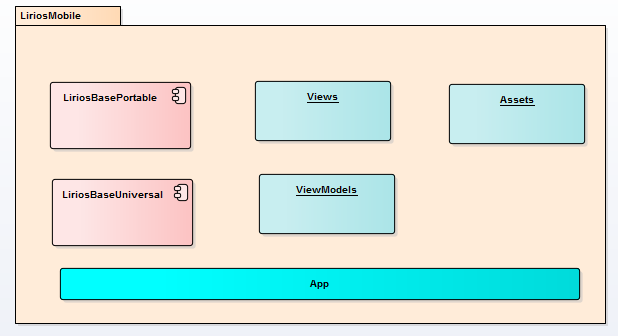


Figura 4‑3: Arquitectura LiriosMobile

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

* **LiriosBasePortable**: Librería que contiene las clases base de la lógica de negocio de la aplicación compatible con aplicaciones web, escritorio y móvil.
* **LiriosBaseUniversal**: Contiene las clases base propias de la aplicación compatible solo con aplicaciones móviles.
* **Views**: Constituyen la interfaz gráfica de usuario de las distintas pantallas de la aplicación.
* **ViewsModels**: Conjunto de clases encargadas de manejar el estado de presentación de los datos en la vista.
* **Assets**: Imágenes, gráficos de la aplicación.
* **App:** Punto de partida del punto de ejecución de la aplicación.

**Arquitectura LiriosWebService**

La arquitectura de LiriosWebService se puede observar en la figura 4-4, la misma que posee los siguientes elementos:

* **LiriosBaseProtable**: Librería que contiene las clases base de la lógica de negocio de la aplicación compatible con aplicaciones web, escritorio y móvil.
* **LiriosBase**: Contiene las clases base propias de la aplicación compatible solo con aplicaciones web y escritorio.
* **DataAccess**: Clases que se encargan del manejo de los datos y comunicarse con la base datos.
* **IMainService**: Interfaz que posee la declaración de los métodos del servicio web.
* **MainService**: Implementación de los métodos de la interfaz.

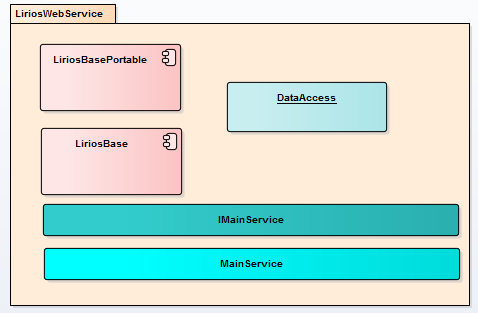


Figura 4‑4: Diagrama de componentes LiriosWebService

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

**Diseño de la base de datos LiriosFadminDB**

En base a los requerimientos y reuniones previas con el product owner se determinó la información que era necesaria almacenar en la base de datos, se procedió al diseño de la misma en un modelo físico. En la Figura 4‑5 se muestra el diagrama de base de datos.



**Figura 4‑5:** Modelo físico de la base de datos LiriosFadminDB

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

**Interfaces de Usuario**

Para LiriosDesktop:

Se utilizaron los colores que vienen por defecto del .Netframework 4.5:





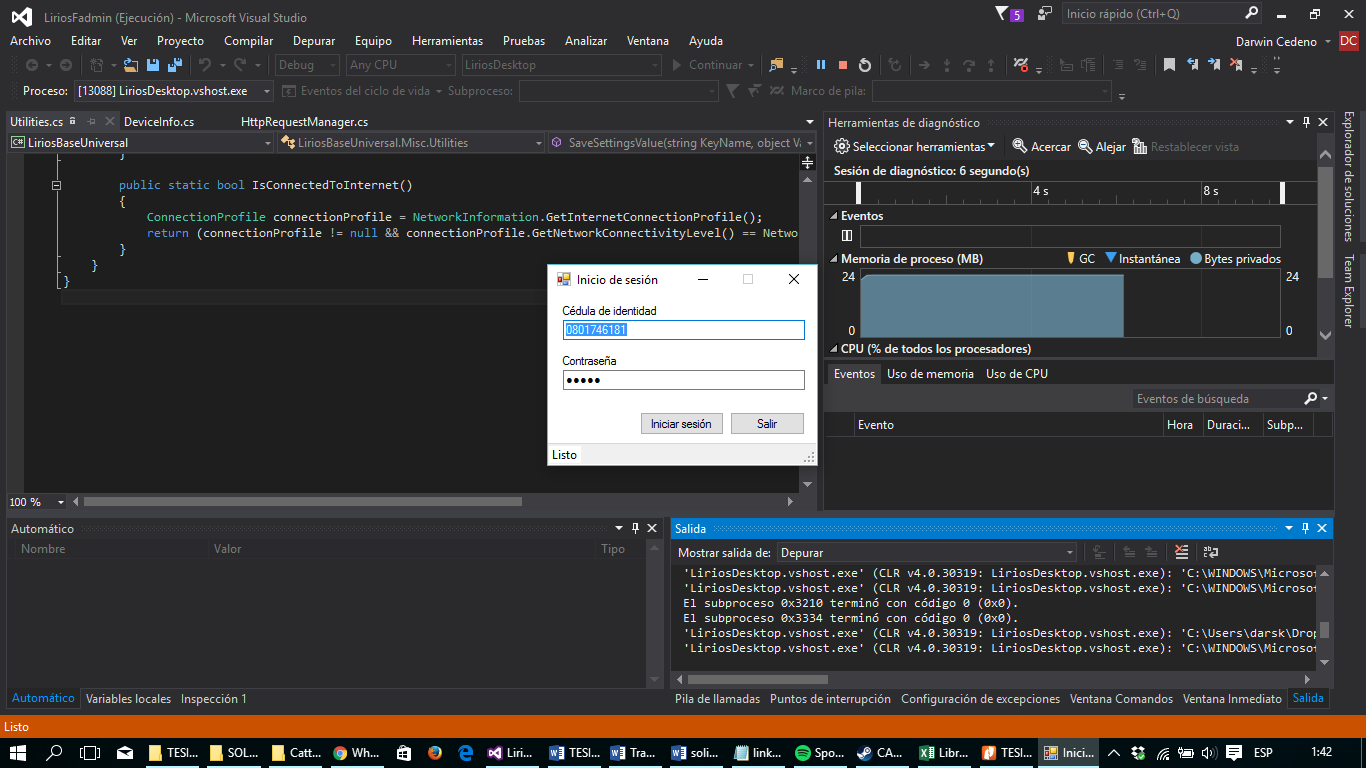


Figura 4‑6: Inicio de sesión LiriosDesktop

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2016

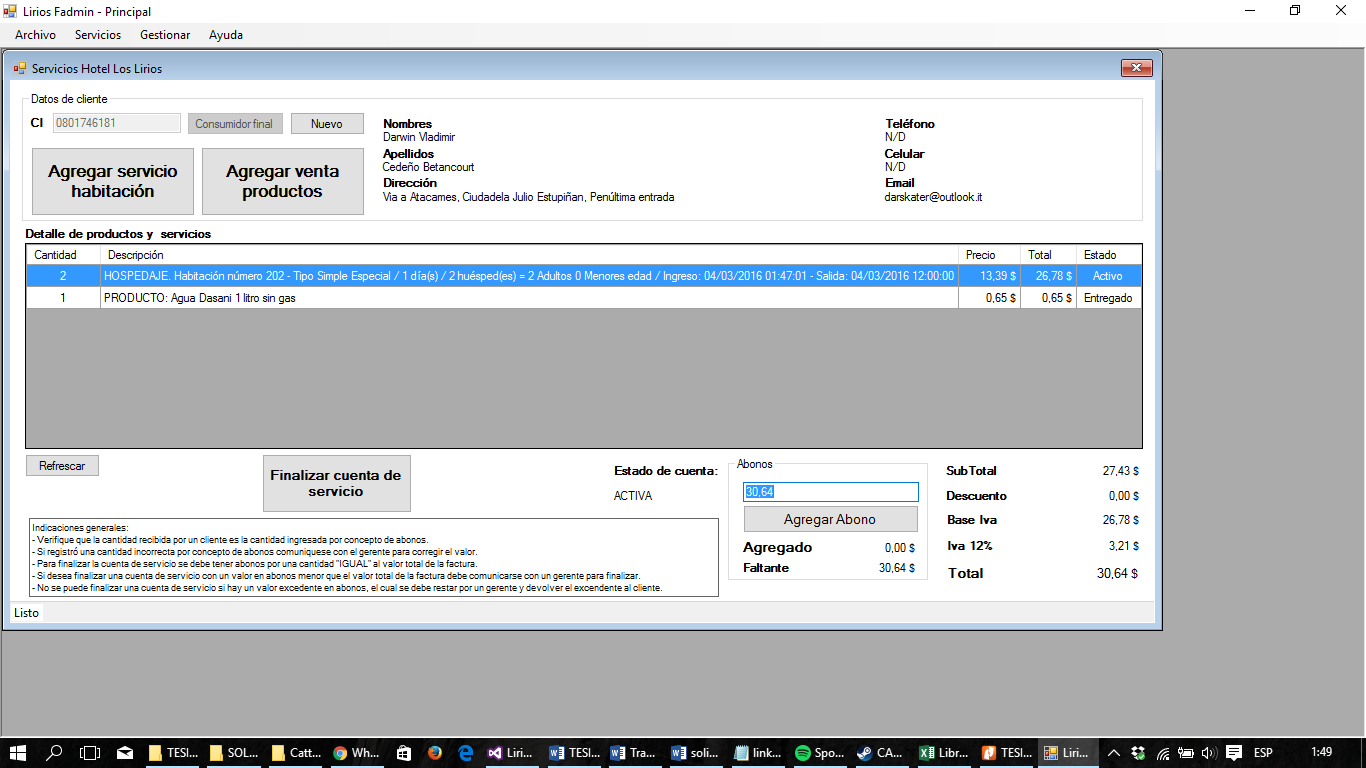


Figura 4‑7: Detalle de productos y servicio

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2016

Para la aplicación móvil LiriosMobile:

Se utilizó la guía estándar de colores de google (*Color - Style Google desing guidelines, 2016):*

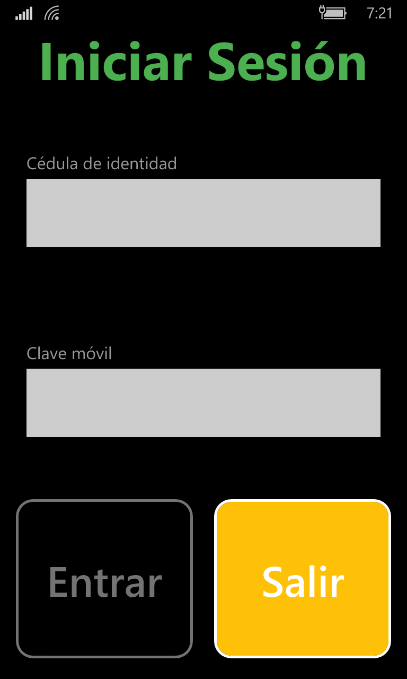
****

Figura 4‑8: Inicio de sesión aplicación móvil

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2016

**Codificación**

Para todo el proyecto se emplea los estándares de codificación del lenguaje C# (Microsoft coding conventions, 2015) para todo lo que concierne a código de programación. Para las interfaces gráficas no se define ningún estándar utilizando solo el estilo que proporciona el IDE de desarrollo. La programación tanto de LiriosDesktop como LiriosMobile en Windows Phone se la realizó con el lenguaje de programación C# y el lenguaje de marcado XAML del IDE Visual Studio Community y para LiriosWebService se utilizó el lenguaje de programación C# con los lenguaje de marcado XML y HTML.

**Sprint 2**

En el sprint 2 se prosigue el desarrollo de las funciones que contiene LiriosDesktop. En la Tabla 4-6 se detallan las historias de usuario escogidas para este sprint con un total de 8.

Tabla 4‑6: Sprint Backlog–Sprint 2- Lirios Desktop

|  |  |
| --- | --- |
| SPRINT 2 | |
| HU7 | Registrar un tipo de habitación |
| HU8 | Actualizar un tipo de habitación |
| HU9 | Eliminar un tipo de habitación |
| HU10 | Registrar una habitación |
| HU11 | Registrar un producto |
| HU12 | Actualizar un producto |
| HU13 | Registro de un cliente |
| HU14 | Actualizar un cliente |

FUENTE: Darwin Cedeño, Diego Quinatoa 2015.

**Sprint 3**

En el sprint 3 se prosigue el desarrollo de las funciones que contiene LiriosDesktop. En la Tabla 4-7 se detallan las historias de usuario escogidas para este sprint con un total de 9.

Tabla 4‑7: Sprint Backlog–Sprint 3 - Lirios Desktop

|  |  |
| --- | --- |
| SPRINT 3 | |
| HU15 | Crear una cuenta de servicio |
| HU16 | Agregar un servicio de hospedaje |
| HU17 | Agregar una venta de un producto |
| HU18 | Finalizar un servicio de hospedaje |
| HU19 | Cancelación de un servicio de hospedaje |
| HU20 | Aplicar descuento del 100% |
| HU21 | Agregar abonos |
| HU22 | Finalizar cuenta de servicio |
| HU23 | Visualizar detalle de la cuenta de servicios |

FUENTE: Darwin Cedeño, Diego Quinatoa 2015.

**Sprint 4**

En el sprint 4 se inicia el desarrollo de la parte móvil (LiriosMobile) del sistema y se finalizan las funciones de LiriosDesktop. En la Tabla 4-8 se detallan las historias de usuario escogidas para este sprint con un total de 10. En este sprint se finaliza el desarrollo de toda la solución LiriosFadmin.

Tabla 4‑8: Sprint Backlog–Sprint 4 Lirios Mobile - Lirios Desktop

|  |  |
| --- | --- |
| SPRINT 4 | |
| HUM1 | Inicio de sesión Cliente |
| HUM2 | Realizar pedido a la habitación |
| HUM3 | Consulta de pedidos realizados |
| HUM4 | Detalle de la cuenta de servicios |
| HUM5 | Inicio de sesión camarero |
| HUM6 | Entrega de un producto |
| HU24 | Visualizar listado de habitaciones |
| HU25 | Actualizar el estado de una habitación |
| HU26 | Notificaciones recepcionista |
| HT6 | Manual de usuario |

FUENTE: Darwin Cedeño, Diego Quinatoa 2015.

**Burn down chart**

En el desarrollo del proyecto se comenzó con 137 puntos de estimación total entre 4 sprints y se llegó a culminar el proyecto con 0 puntos de estimación pendientes. El promedio de puntos de estimación entre Sprints es de 34.25 puntos de estimación. El Sprint con mayor número de puntos de estimación es el Sprint 3 (40). En la Figura 4-6 se muestra el gráfico de Burn down chart:

Figura 4‑9: Diagrama de Burn Down

**FUENTE**: Darwin Cedeño y Diego Quinatoa 2015

Una vez terminado el proyecto se obtuvo lo siguiente:

- 37 historias de usuario.

- 4 Sprints, de 2 semana cada uno, de 5 días laborables a 8 horas de trabajo diario.

- 67 pruebas de aceptación (historias de usuario y tareas de ingeniería de las historias de usuario).

- 2 Aplicaciones (LiriosDesktop y LiriosMobile Windows Phone).

- 1 aplicación web (LiriosWebService).

- 1 Base de datos (LiriosFadminDB).

# CONCLUSIONES

Con la culminación del trabajo de investigación “Análisis comparativo entre los RDBMS ligeros SQLite y SQL server compact en Windows Phone para el sistema de facturación del hotel los lirios” se obtuvieron las siguientes conclusiones:

* El análisis comparativo entre los RDBMS ligeros SQLite y SQL Server Compact, determinó que SQLite brinda un mayor porcentaje de eficiencia en el consumo de recursos hardware con el 96,09 % con relación a SQL Server Compact con el 60,95 %.
* Al ejecutar los prototipos desarrollados, SQLite resultó ser menos eficiente en recurso hardware al medir el tiempo del uso CPU en la prueba de eliminación de datos con el 24,37 %.
* A pesar de que SQL Server Compact viene incluido en el sistema Windows Phone al realizar las pruebas de Insert, Update, Delete, Select, este es menos eficiente en recurso hardware ya que consume mucho más recursos hardware que SQLite que es una librería externa.
* Con la implementación del sistema de facturación del hotel “Los Lirios”, se logró automatizar el proceso de facturación, gestión de pedidos y administración de servicios con el fin de brindar una buena atención al cliente, siendo uno de los primeros hoteles en la ciudad del Tena en incorporar a sus servicios una aplicación móvil.

# RECOMENDACIONES

A futuro se recomienda:

* Utilizar SQLite como RDBMS ligero en el desarrollo de futuras aplicaciones móviles ya que SQL Server Compact ya no tiene soporte por Microsoft para la nueva plataforma Windows 10.
* Utilizar una metodología ágil para el desarrollo de un proyecto de software, por las ventajas que estas brindan tales como: la optimización del tiempo, organización del equipo de trabajo, aumento de la calidad del producto software.
* Integrar en LiriosMobile las notificaciones PUSH para enviar a los huéspedes avisos referentes a promociones, noticias, descuentos y consumos.
* Desarrollar LiriosMobile para IOS y Android ya que en el mercado actual de aplicaciones estas plataformas lideran.
* Autorizar el sistema por el SRI para darle legalidad al proceso de facturación y el sistema pueda ser implementado en el hotel.
* Implementar otros servicios que ofrece el hotel en el módulo móvil tales como: la gestión de reservas, el servicio de restaurante entre otros a futuro.
* Agregar reportes al sistema de facturación ya que por el momento el sistema solo comprende el proceso de facturación y gestión de pedidos sin esta característica.

# 

# BIBLIOGRAFÍA

1. **ANDRÉS RODRIGUEZ**, *¿Qué es Windows Phone?,* [en línea], Colombia – Pereira: Universidad de Pereira, 2013, p.17, [Consulta 12 de Abril 2015].

Disponible en:

<http://ribuc.ucp.edu.co:8080/jspui/bitstream/handle/10785/1729/CDMIST66.pdf?sequence=2>

1. **ANDRADE ARRIETA, Mario Fernando, & PARRA GARCÍA, Jhony Darío**. *Análisis de rendimiento entre postgresql y sql server usando hammerdb y manage engine aplicado al sistema académico de conduespoch* (tesis) (pregrado). Escuela Superior Politécnica del Chimborazo, Facultad de Informática y Electrónica, Escuela de Ingeniería en Sistemas, Riobamba, Ecuador, 2014. P.23.
2. **ANTONIO CABERO,** *Historia de SQLite,* [en línea], s.f. [Consulta 22 de Abril 2015].

Disponible en:

<https://www.scribd.com/fullscreen/52882068?access_key=key-7tv13o05cj4dj11lq9b&allow_share=true&escape=false&view_mode=scroll>

1. *Acerca de SQLite (en inglés),* [en línea], s.f. [Consulta 22 de Abril 2015].

Disponible en:

[http://sqlite.org/about.html](http://sqlite.org/about.html%20)

1. **ADRIANA PERALTA,** *Metodología SCRUM,* [en línea]. Montevideo-Uruguay: Universidad ORT Uruguay, 2003, pp 2-4, [Consulta 10 de junio 2015].

Disponible en:

<http://fi.ort.edu.uy/innovaportal/file/2021/1/scrum.pdf>

1. *App memory limits for Windows Phone,* [en línea, en inglés], Microsoft 2015. [Consulta 25 de Abril 2015].

Disponible en:

<https://msdn.microsoft.com/en-us/library/windows/apps/jj681682(v=vs.105).aspx>

1. *Arquitectura de SQLite (en inglés),* [en línea], 2015. [Consulta 22 de Abril 2015].

Disponible en:

<http://sqlite.org/arch.html>

1. *Bases de datos local para Windows Phone - Panorama arquitectónico (en inglés),* [en línea], MSDN Microsoft, 2015. [Consulta 19 de Abril 2015].

Disponible en:

<https://msdn.microsoft.com/library/windows/apps/hh202860(v=vs.105).aspx>

1. *Características de SQLite (en inglés),* [en línea], 2015 [Consulta 22 de Abril 2015].

Disponible en:

<http://sqlite.org/features.html>

1. *C# Coding Conventions (C# Programming Guide),* [En línea],Microsoft, 2015. [Consulta 10 de junio 2015].  
   Disponible en:

<https://msdn.microsoft.com/en-us/library/ff926074.aspx>

1. **DIEGO PINO,** *Sistemas de gestión de bases de datos relaciones,* [blog]. Marzo 2009. [Consulta 15 de Mayo 2015].

Disponible en:

<http://diegopino.blogspot.com/2009/03/los-sistemas-de-bases-de-datos.html>

1. **FILEIN ROMMEL,** *Las bases de datos embebidas,* [en línea]. SG BUZZ, Septiembre – Octubre 2007. [Consulta 19 de mayo 2015].

Disponible en:

[http://sg.com.mx/revista/17/sqlite-la-base-datos-embebida#.VX9cP](http://sg.com.mx/revista/17/sqlite-la-base-datos-embebida%23.VX9cP)

1. *Guía de aplicaciones de la plataforma universal de Windows,* [en línea]. MSDN Microsoft, 2015. [Consulta 18 de mayo 2015].

Disponible en:

[https://msdn.microsoft.com/es-es/library/windows/apps/dn894631.aspx](https://msdn.microsoft.com/es-es/library/windows/apps/dn894631.aspx%20)

1. *Color - Style Google desing guidelines,* [en línea]. Google, 2016. [Consulta 3 de Marzo 2015].

Disponible en:

[https://www.google.com/design/spec/style/color.html#color-themes](https://www.google.com/design/spec/style/color.html%23color-themes)

1. **IDC (International Data Company),** *Smartphone OS market share,* [en línea], USA: IDC, 2015. [Consulta 10 de Abril 2015].

Disponible en:

[http://www.idc.com/prodserv/smartphone-os-market-share.jsp](http://www.idc.com/prodserv/smartphone-os-market-share.jsp%20)

1. *Introducción a la arquitectura de Windows Phone,* [en línea]. USA-Seattle: MSDN Microsoft, Julio 2015. [Consulta 18 de mayo 2015].

Disponible en:

<https://sysdev.microsoft.com/en-us/Hardware/oem/docs/Getting_Started/Windows_Phone_architecture_overview>

1. *Instalación de SQLite en Windows Phone – Microsoft Open Technologies (en inglés),* [en línea], 2015. [Consulta 22 de Abril 2015].

Disponible en:

<https://sqlitepcl.codeplex.com/documentation>

1. IEEE, 1992. *IEEE 1045-1992, Standard for Software Productivity Metrics.* [En línea]. [Consulta 22 de Abril 2015]   
   Disponible en: <http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=211732>
2. IEEE, 1998. *830-1998 - IEEE Recommended Practice for Software Requirements Specifications.* [En línea]. [Consulta 22 Abril 2015]   
   Disponible en: <https://standards.ieee.org/findstds/standard/830-1998.html>
3. **JUAN GABRIEL OSORIO,** *Las bases de datos incrustadas,* [en línea]. Colombia-Medellín: Universidad autónoma latinoamericana (Unaula), SlideShare, Noviembre 2010, p.2. [Consulta 10 de mayo 2015].

Disponible en:

<http://es.slideshare.net/inspirateunaula/bases-de-datos-embebidas-5925744>

1. **LEANDRO ALEGSA,** *Definición de SDK,* [en línea]. Argentina-Santa fe: Alegsa, Mayo 2010. [Consulta 15 de mayo 2015].

Disponible en:

<http://www.alegsa.com.ar/Dic/sdk.php>

1. **LEANDRO ALEGSA,** *Definición de Eficiencia,* [en línea]. Argentina-Santa fe: Alegsa, Mayo 2010. [Consulta 15 de mayo 2015].

Disponible en:

<http://www.alegsa.com.ar/Dic/eficiencia.php>

1. **LETELIER, PATRICIO. & PENADÉS, Ma CARMEN** *Metodologías ágiles para el desarrollo de Software, eXtreme Programming (XP),* [en línea]*,* s.f*,*  [Consulta 18 de Mayo 2015]

Disponible en:

<http://www.cyta.com.ar/ta0502/v5n2a1.htm>

1. *Lo nuevo de SQL Server Compact Edition,* [en línea]. MSDN Microsoft, 2015. [Consulta 18 de mayo 2015].

Disponible en:

<https://msdn.microsoft.com/es-es/library/ayee3tzx(v=vs.85).aspx>

1. *LINQ to SQL (SQL Server Compact), [en línea], Microsoft, 2015. [Consulta 18 de Abril 2015].*

Disponible en:

<https://technet.microsoft.com/es-es/library/bb896144(v=sql.105).aspx>

1. *Microsoft C# coding conventions, [en línea], Microsoft, 2015. [Consulta 03 de Marzo 2015].*

Disponible en:

<https://msdn.microsoft.com/en-us/library/ff926074.aspx.>

1. *Novedades y características de Windows Phone, [en línea], MSDN Microsoft, 2015. [Consulta 13 de Abril 2015].*

Disponible en:

<https://msdn.microsoft.com/es-es/library/windows/apps/dn632424.aspx>

1. **PETE DEEMER & GABRIELLE BENEFIELD & CRAIG LARMAN & BAS VODDE,** *metodología ágil SCRUM,* [en línea], 2012. [Consulta 12 de abril 2015].

Disponible en:

<http://www.scrumprimer.org/primers/es_scrumprimer20.pdf>

1. *Definición de pruebas de normalidad,* [en línea], 2015. [Consulta 04 de marzo 2016].

Disponible en:

<http://www.manufacturingterms.com/Spanish/Normality-Tests.html>

1. *Procesos y roles SCRUM, [en línea], s.f. [Consulta 22 de Abril 2015].*

*Disponible en:*

<https://www.softeng.es/es-es/empresa/metodologias-de-trabajo/metodologia-scrum/proceso-roles-de-scrum.html>

1. **RICARDO J,** *Desarrollo de aplicaciones para Windows Phone,* [blog]. MSDN blogs, julio 2012. [Consulta 13 de Abril 2015].

Disponible en:

<http://blogs.msdn.com/b/ricardoj/archive/2012/07/12/desarrollo-de-aplicaciones-para-windows-phone-herramientas.aspx>

1. **SERGIO GALLARDO SALES,** *Las Universal Apps,* [blog]. MSDN Microsoft blogs España: Abril 2014. [Consulta 14 de Abril 2015].

Disponible en:

<http://blogs.msdn.com/b/esmsdn/archive/2014/04/09/universal-apps-un-mismo-c-243-digo-para-windows-8-1-y-windows-phone-8-1.aspx>

1. *Sentencias y lenguaje de consulta SQL en SQLite,* [en línea, en inglés], SQLite org, 2015. [Consulta 20 de Abril 2015].

Disponible en:

<https://www.sqlite.org/lang.html>

1. *Silverlight for Windows Phone, Microsoft 2015*[Consulta 13 de Abril 2015].

Disponible en:

<https://msdn.microsoft.com/es-es/library/ff426934(v=vs.95).aspx>

1. *Agile User Stories.* [en línea], Scrum Alliance, 2013. [Consulta 25 de abril 2015].   
   Disponible en: <https://www.scrumalliance.org/community/articles/2013/september/agile-user-stories>
2. *Tipos de datos SQL Server Compact,* [en línea], MSDN Microsoft, 2015. [Consulta 19 de Abril 2015].

Disponible en:

<https://technet.microsoft.com/es-ec/library/ms172424(v=sql.105).aspx>

1. *Tipos de datos de SQLite (en inglés),* [en línea], 2015. [Consulta 22 de Abril 2015].

Disponible en:

<https://www.sqlite.org/datatype3.html>

1. *Ventajas y limitaciones de SQL Server Compact,* [en línea], 2015. [Consulta 25 de Abril 2015].

Disponible en:

<https://grimpidev.wordpress.com/2008/10/24/sqlite-vs-sql-server-compact-edition-parte-1/>

1. *Ventajas y limitaciones de SQLite (en inglés),* [en línea], 2015. [Consulta 25 de Abril 2015].

Disponible en:

<http://sqlite.org/about.html>

# ANEXOS

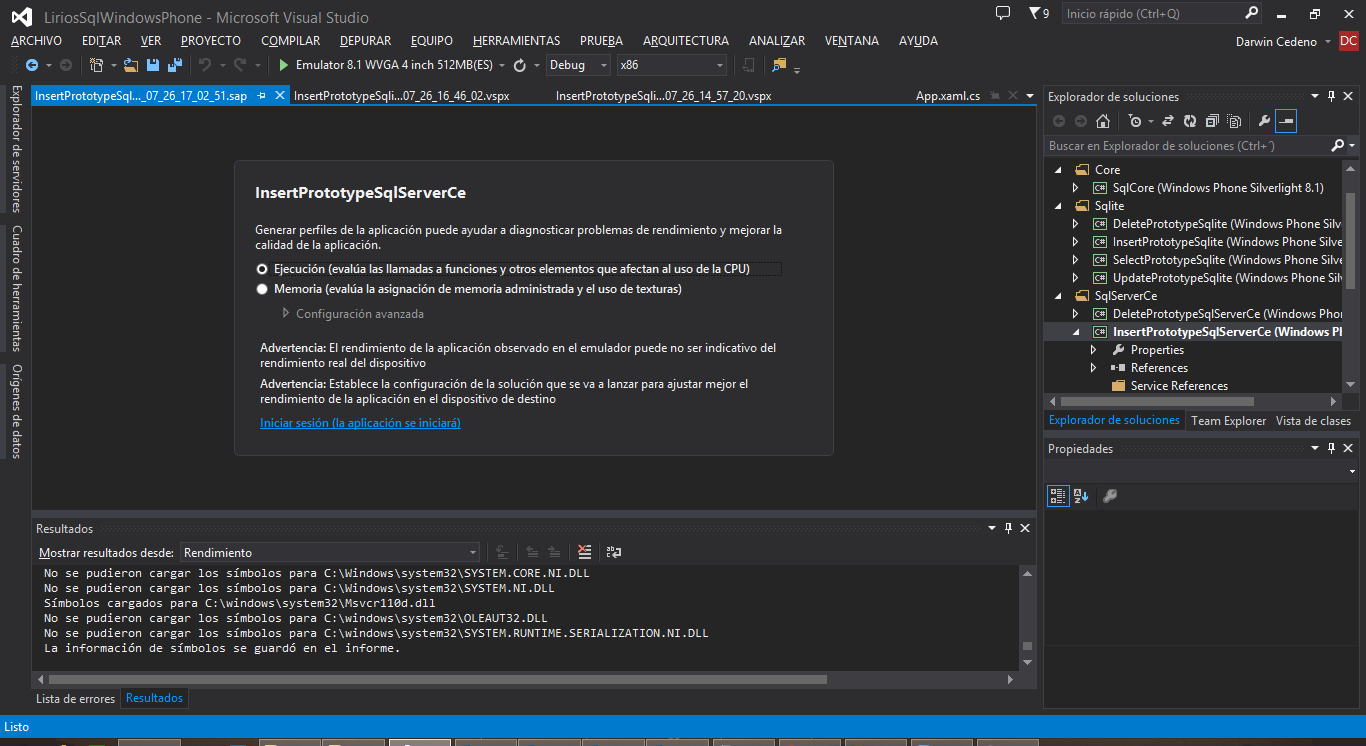
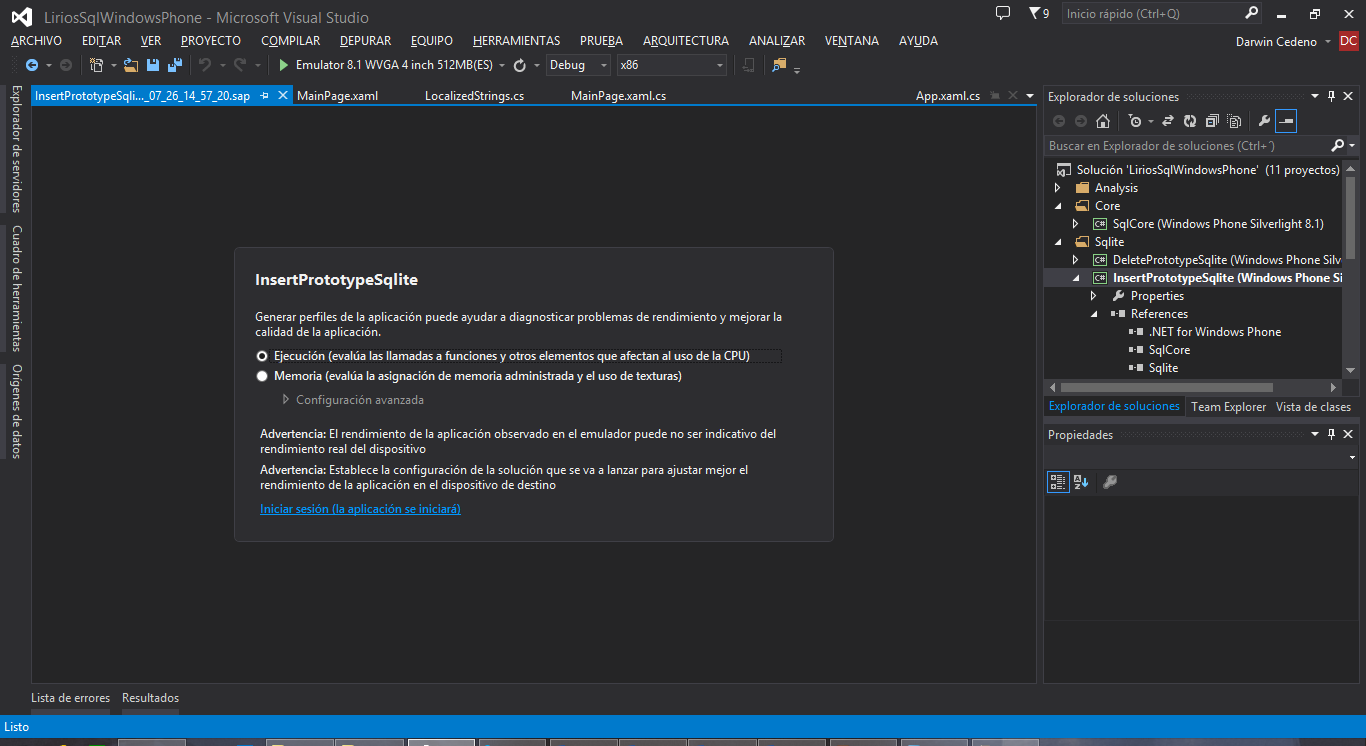
## ANEXO A: Prototipos Desarrollados para los RDMBS ligeros SQLite y SQL Server Compact.

### Prototipo INSERT (SQLite / SQL Server Compact Edition)

* **Herramienta Profiling Tools Visual Studio**

En Visual Studio se presiona las teclas Alt + F1 o clic en el menú DEPURAR y se escoge la opción “Iniciar el análisis de aplicación de Windows Phone”.

Aparecerá la siguiente pantalla como indica la figura 1-A1, de acuerdo al prototipo con el que se está trabajando:



Herramienta Profiling Tools de Visual Studio, Prototipos INSERT SQLite, SQL Server Compact.

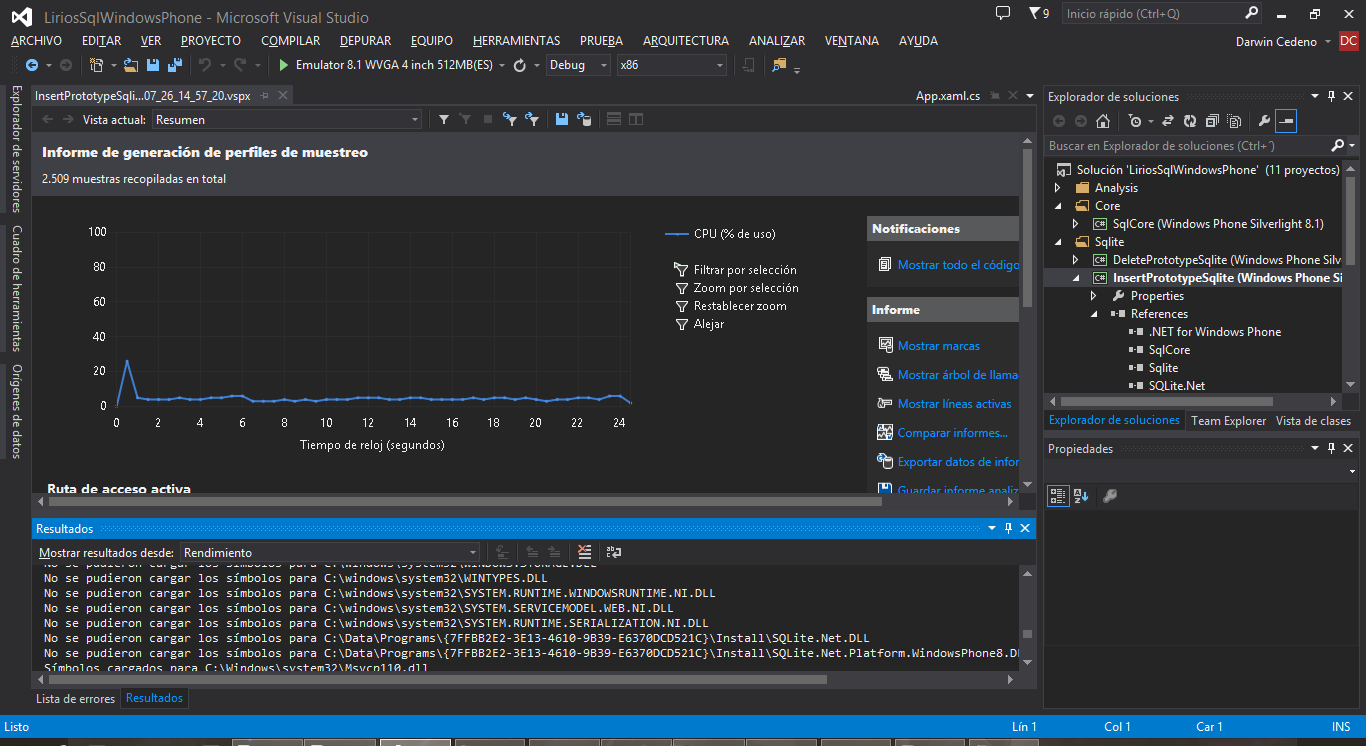
**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

Se escoge la opción Ejecución e iniciar sesión, con esta acción la aplicación se ejecutará y será monitoreada por la herramienta de perfil.

Durante el proceso de ejecución del prototipo, este realiza los siguientes pasos:

* Verificar si existe el archivo de base de datos, en caso de ser afirmativo lo elimina.
* Crear un nuevo archivo de base de datos.
* Insertar 1200 registros con datos aleatorios, a excepción de la clave primaria q es un auto numérico.
* Capturar el pico máximo de memoria utilizado por el prototipo y el tamaño del archivo de base de datos resultante. Estos valores se los almacena en la carpeta local del almacenamiento aislado de la aplicación.
* Forzar la terminación de la ejecución de la aplicación mediante código.

Una vez que la aplicación termina su ejecución, la herramienta de profiling inicia a consolidar los datos capturados y los muestra como se ve en la siguiente figura 2-A1:



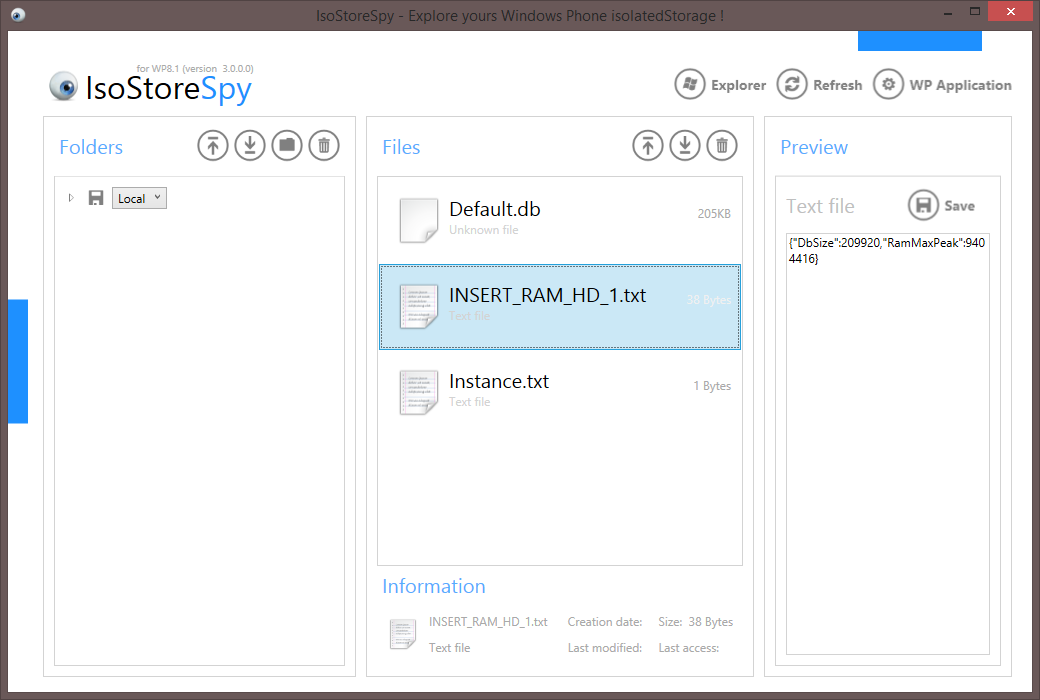
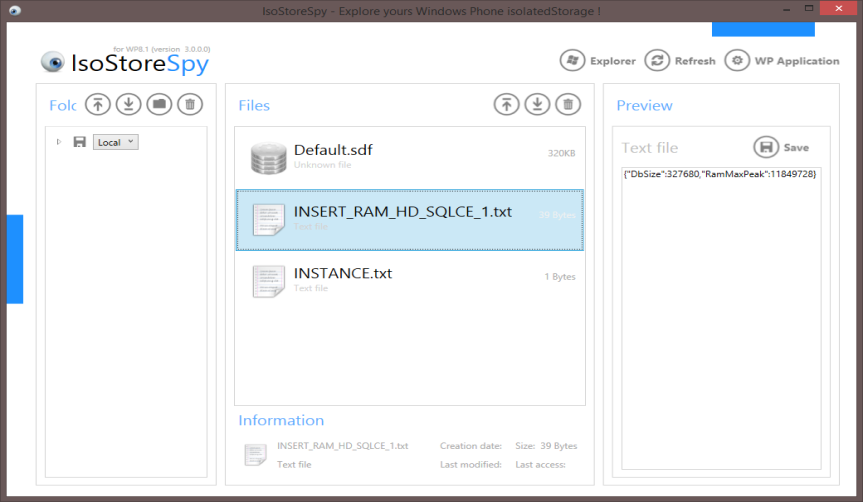
Reporte de la herramienta de profiling con los datos de utilización del CPU (tiempo y unidades de procesamiento).

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

De este reporte se guarda el archivo de marcas, el cual contiene el tiempo de ejecución de la aplicación y las unidades de procesamiento utilizadas de CPU.

* **Almacenamiento aislado - IsoStoreSpy**

Para poder acceder al almacenamiento aislado de los archivos del prototipo de aplicativo móvil, se utiliza la herramienta IsoStoreSpy con la que se extrae el archivo de captura con los datos del pico máximo de uso en memoria RAM y tamaño de archivo de base de datos resultante obtenidos durante la ejecución del prototipo. Este archivo tiene por nombre “INSERT\_RAM\_HD\_SQLCE\_<número de ejecución del prototipo>.txt” para SQL Server Compact y “INSERT\_RAM\_HD\_<número de ejecución del prototipo>.txt” para SQLite como se aprecia en la figura 3-A1:



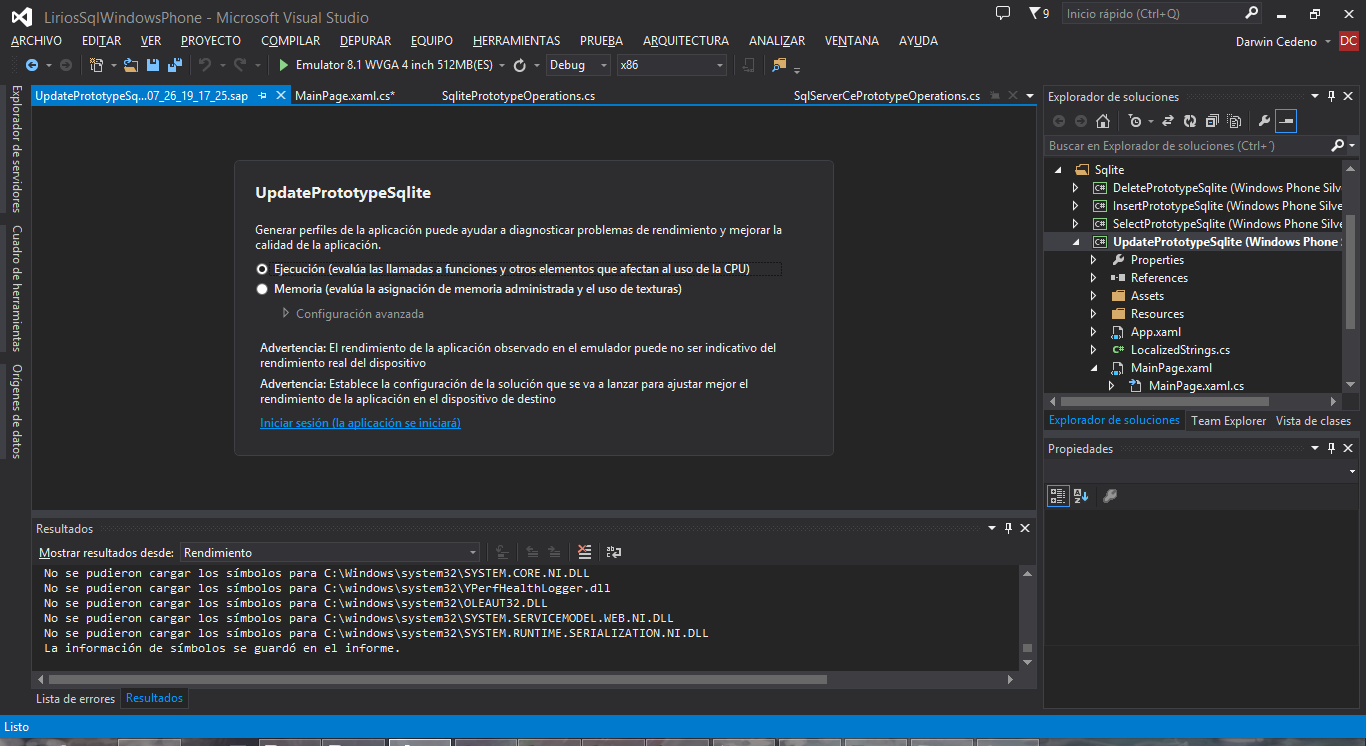
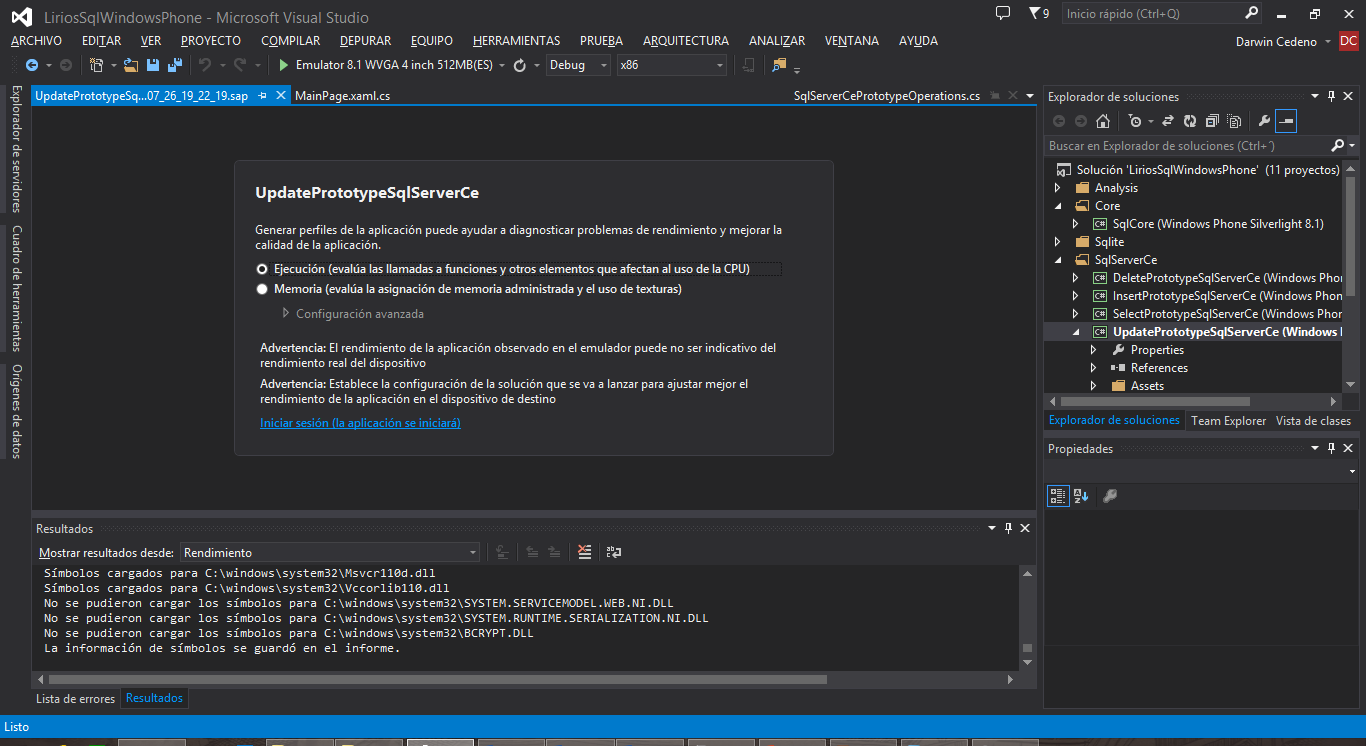
Vista del almacenamiento aislado del prototipo INSERT (SQL Server Compact, SQLite) con la herramienta IsoStoreSpy.

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

### Prototipo UPDATE (SQLite / SQL Server Compact Edition)

En Visual Studio se presionan las teclas Alt + f1 o clic en el menú DEPURAR y se escoge la opción “Iniciar el análisis de aplicación de Windows Phone”.

Aparecerá la siguiente pantalla como indica la figura 4-A2 de acuerdo al prototipo con el que se está trabajando:



Herramienta profiling tolos de Visual Studio, Prototipos UPDATE SQLite, SQL Server Compact.

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

Se escoge la opción ejecución e iniciar sesión, con esta acción la aplicación se ejecutará y será monitoreada por la herramienta de perfil (profiling).

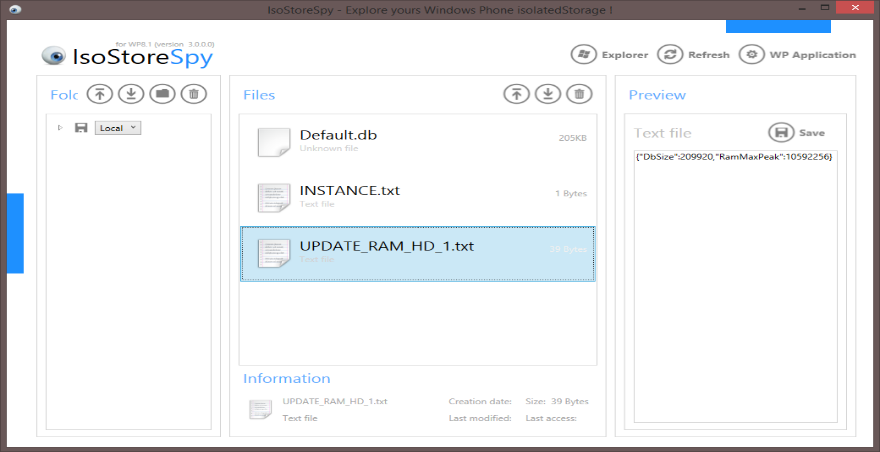
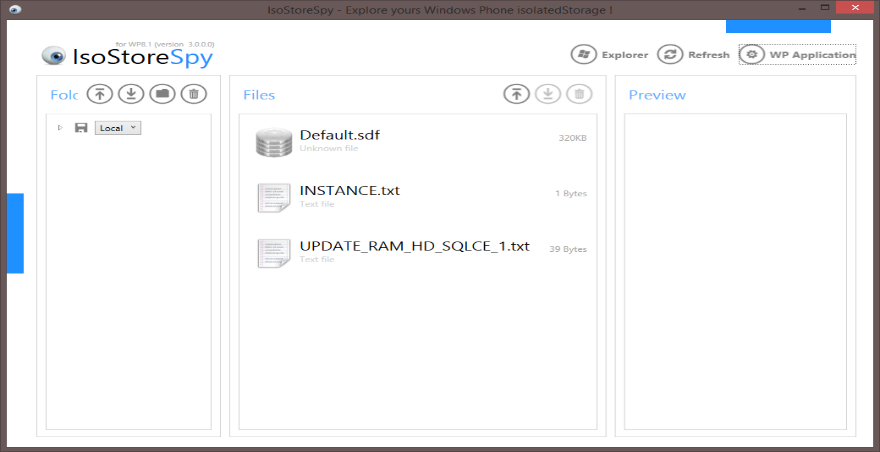
Durante el proceso de ejecución del prototipo, este realiza los siguientes pasos:

* Verificar si existe el archivo de base de datos, en caso de ser afirmativo lo elimina.
* Restaurar un nuevo archivo de BD previamente lleno de datos.
* Actualizar 1200 registros al azar.
* Capturar el pico máximo de memoria utilizado por el prototipo y el tamaño del archivo de base de datos resultante. Estos valores se los almacena en la carpeta local del almacenamiento aislado de la aplicación.
* Forzar la terminación de la ejecución de la aplicación mediante código.

Una vez que la aplicación termina su ejecución, la herramienta de profiling inicia a consolidar los datos capturados y los muestra tal como indica la figura 2-A. De este reporte se guarda el archivo de marcas que contiene el tiempo de ejecución de la aplicación y las unidades de procesamiento utilizadas por le CPU.

* **Almacenamiento aislado - IsoStoreSpy**

Para acceder al almacenamiento aislado del prototipo, se utiliza la herramienta IsoStoreSpy con la que se extrae el archivo de captura con los datos del pico máximo de uso memoria RAM y tamaño de archivo de base de datos resultante durante la ejecución del prototipo. Este archivo tiene por nombre “UPDATE\_RAM\_HD\_SQLCE\_<número de ejecución del prototipo>.txt” para SQL Server Compact y “UPDATE\_RAM\_HD\_<número de ejecución del prototipo>.txt” para SQLite como se ilustra en la figura 5-A2:



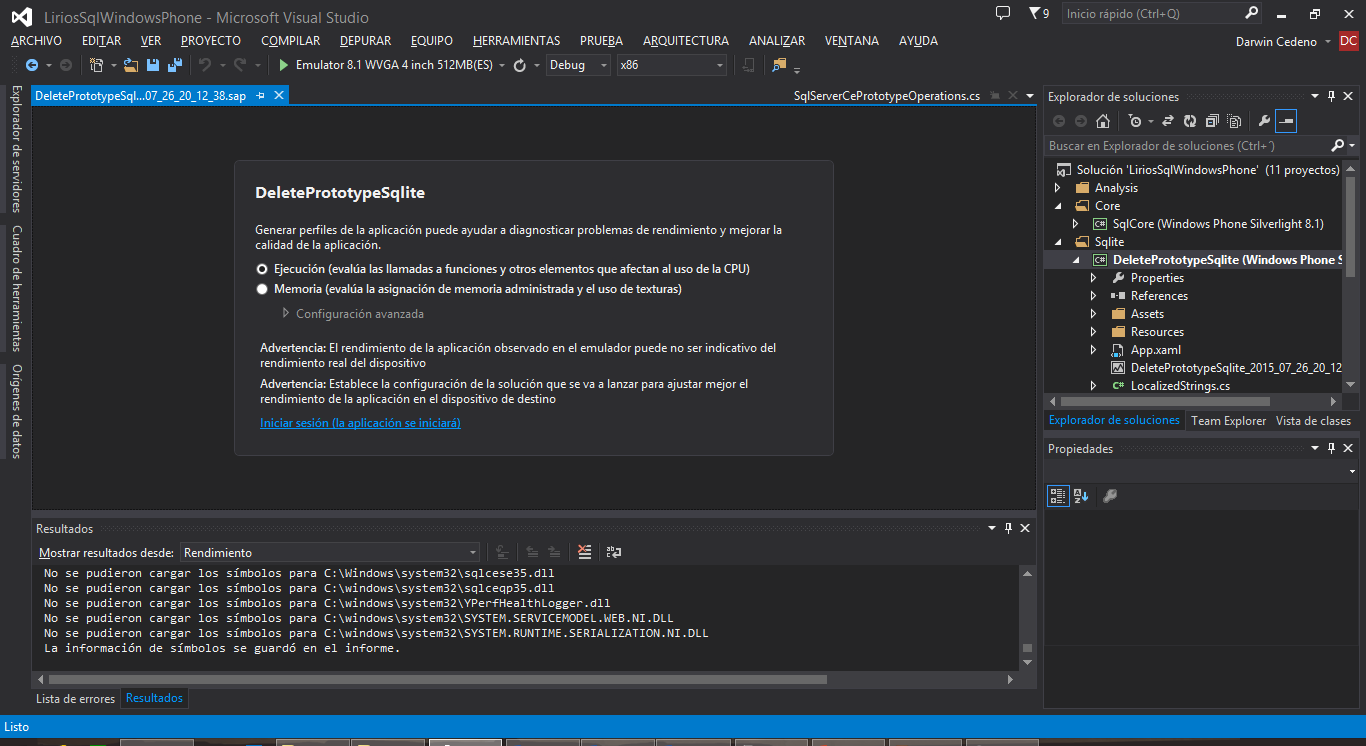
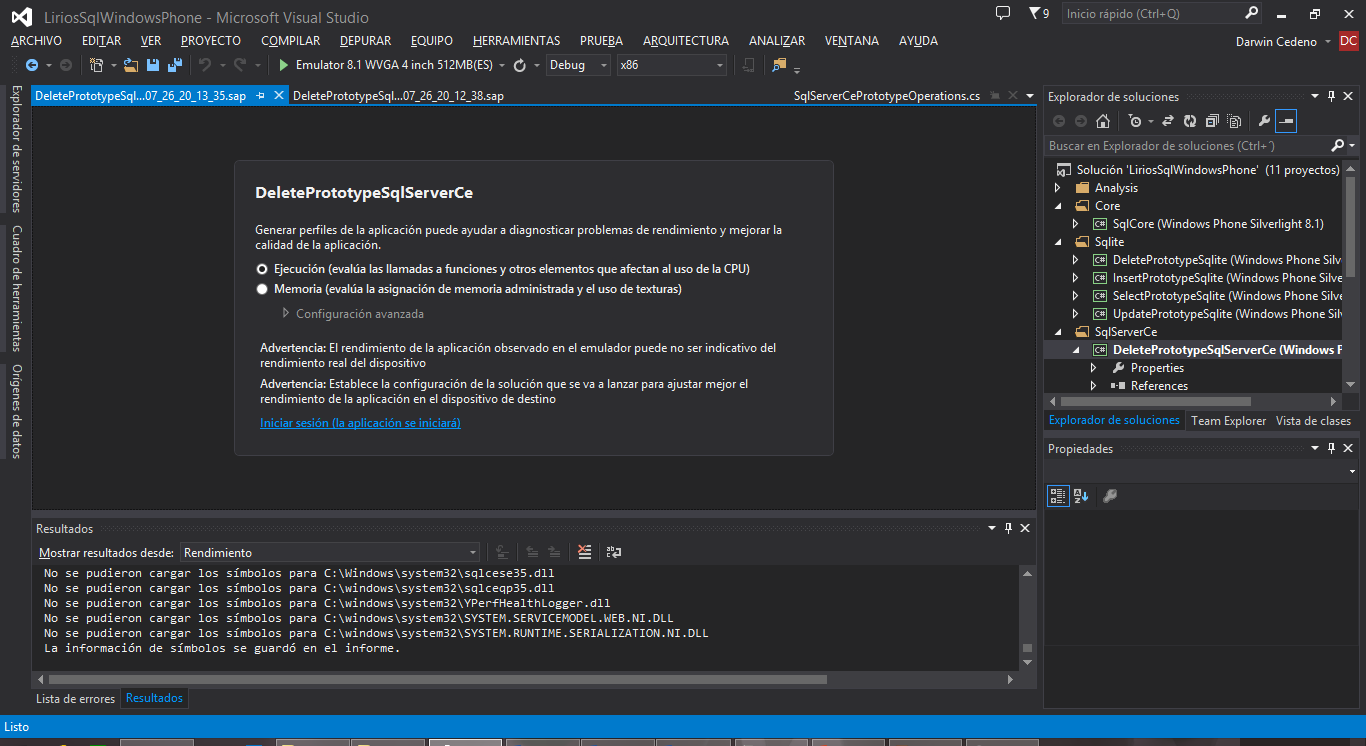
Vista del almacenamiento aislado del prototipo UPDATE (SQL Server Compact, SQLite) con la herramienta IsoStoreSpy.

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

### Prototipo DELETE (SQLite / SQL Server Compact Edition)

En Visual Studio se presionan las teclas Alt + f1 o clic en el menú DEPURAR y se escoge la opción “Iniciar el análisis de aplicación de Windows Phone”.

Aparecerá la siguiente pantalla como indica la figura 6-A2 de acuerdo al prototipo con el que se está trabajando:



Herramienta profiling tools de Visual Studio, prototipo DELETE SQLite, SQL Server Compact.

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

Se escoge la opción Ejecución e iniciar sesión, con esta acción la aplicación se ejecutará y será monitoreada por la herramienta de perfil (profiling).

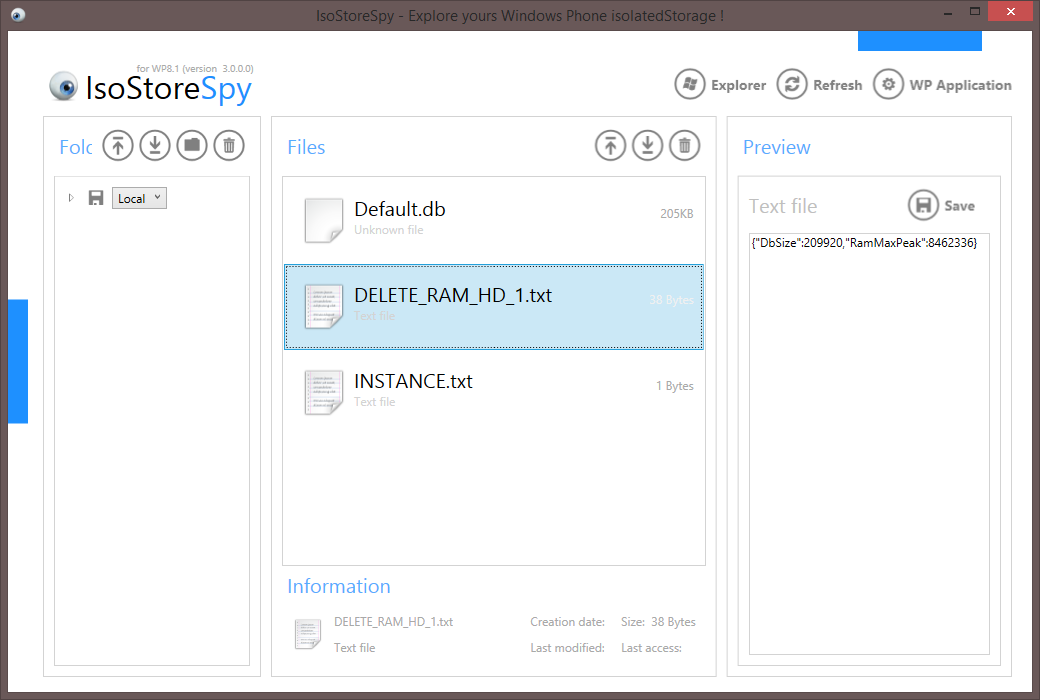
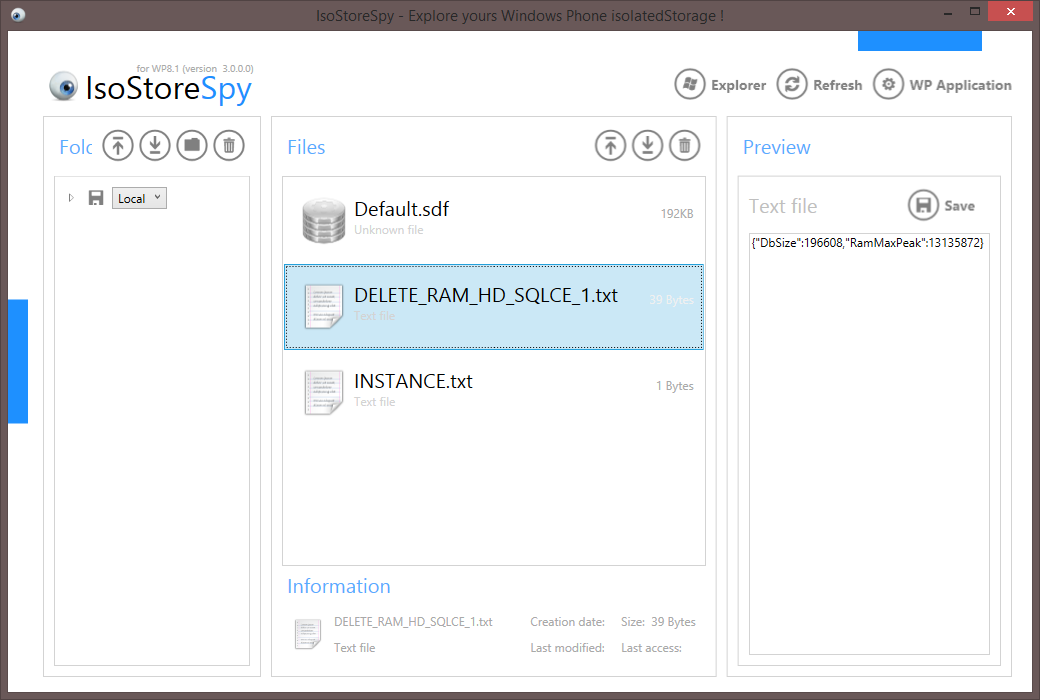
Durante el proceso de ejecución del prototipo, este realiza los siguientes pasos:

* Verificar si existe el archivo de base de datos, en caso de ser afirmativo lo elimina.
* Restaurar un nuevo archivo de BD previamente lleno de datos.
* Elimina los 1200 registros uno a uno en orden.
* Capturar el pico máximo de memoria utilizado por el prototipo y el tamaño del archivo de base de datos resultante. Estos valores se los almacena en la carpeta local del almacenamiento aislado de la aplicación.
* Forzar la terminación de la ejecución de la aplicación mediante código.

Una vez que la aplicación termina su ejecución, la herramienta de profiling inicia a consolidar los datos capturados y los muestra tal como indica la figura 2-A. De este reporte se guarda el archivo de marcas que contiene el tiempo de ejecución de la aplicación y las unidades de procesamiento utilizadas por le CPU.

* **Almacenamiento aislado**

Para acceder al almacenamiento aislado del prototipo, se utiliza la herramienta IsoStoreSpy con la que se extrae el archivo de captura con los datos del pico máximo de uso memoria RAM y tamaño de archivo de base de datos resultante durante la ejecución del prototipo. Este archivo tiene por nombre “DELETE\_RAM\_HD\_SQLCE\_<número de ejecución del prototipo>.txt” para SQL Server Compact y “DELETE\_RAM\_HD\_<número de ejecución del prototipo>.txt” para SQLite como se ilustra en la figura 7-A2:

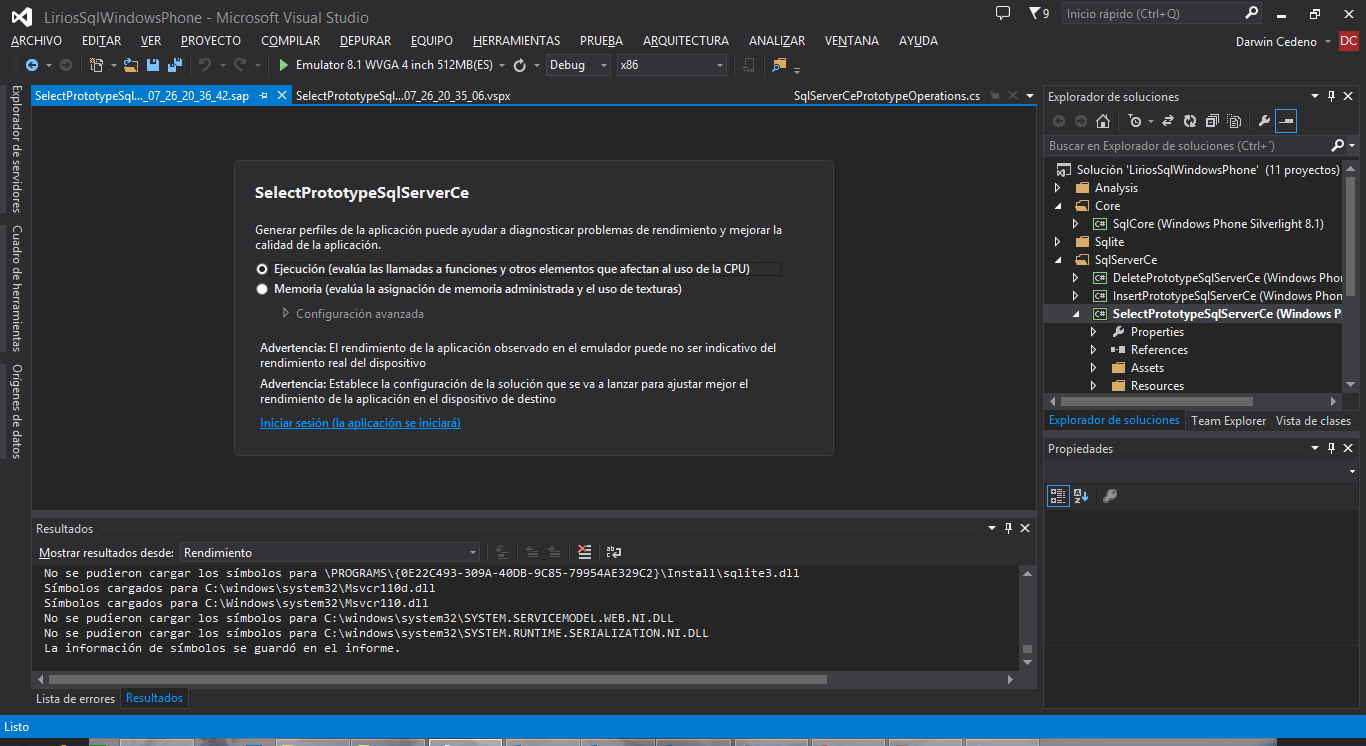
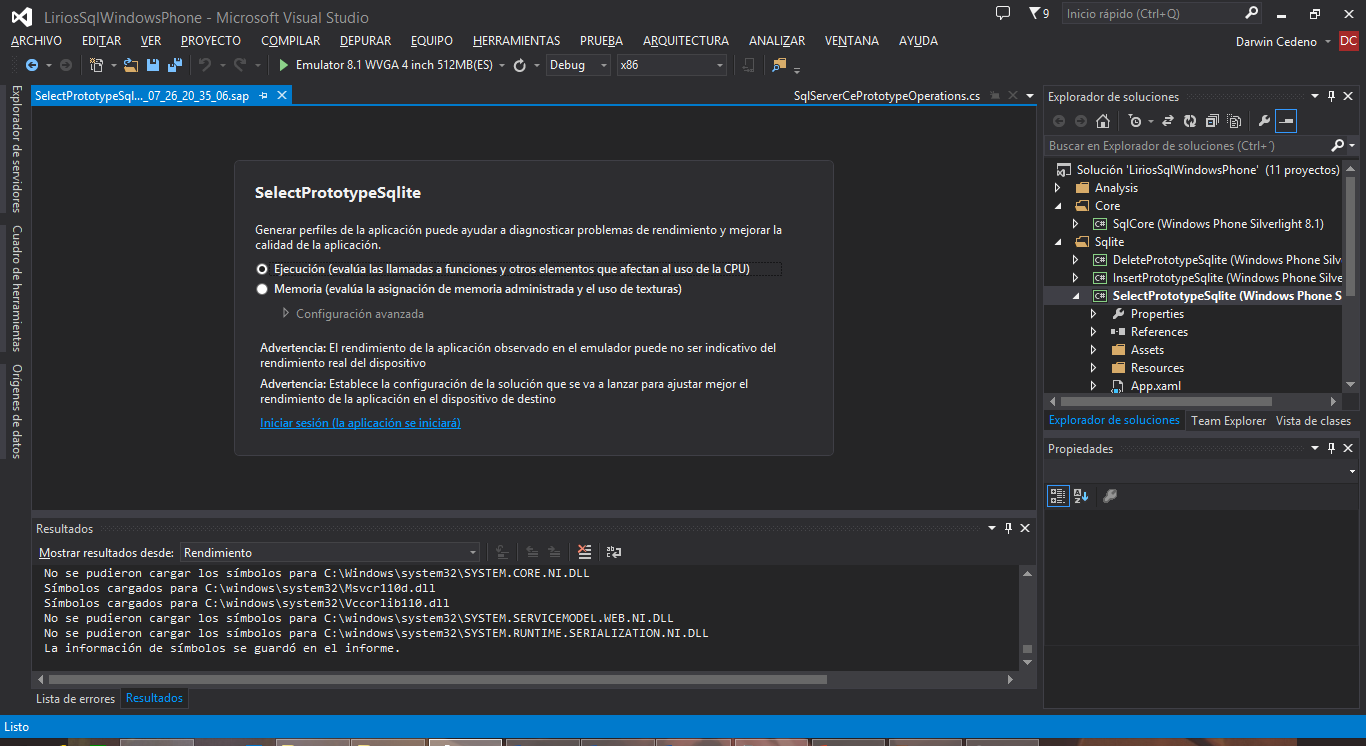


Vista del almacenamiento aislado del prototipo DELETE (SQL Server Compact, SQLite,) con la herramienta IsoStoreSpy.

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

### Prototipo SELECT (SQLite / SQL Server Compact Edition)

En Visual Studio se presionan las teclas Alt + f1 o clic en el menú DEPURAR y se escoge la opción “Iniciar el análisis de aplicación de Windows Phone”. Aparecerá la siguiente pantalla como indica la figura 8-A4 de acuerdo al prototipo con el que se está trabajando:



Herramienta de profiling de Visual Studio, Prototipos SELECT SQLite, SQL Server Compact.

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

Se escoge la opción Ejecución e iniciar sesión, con esta acción la aplicación se ejecutará y será monitoreada por la herramienta de perfil (profiling).

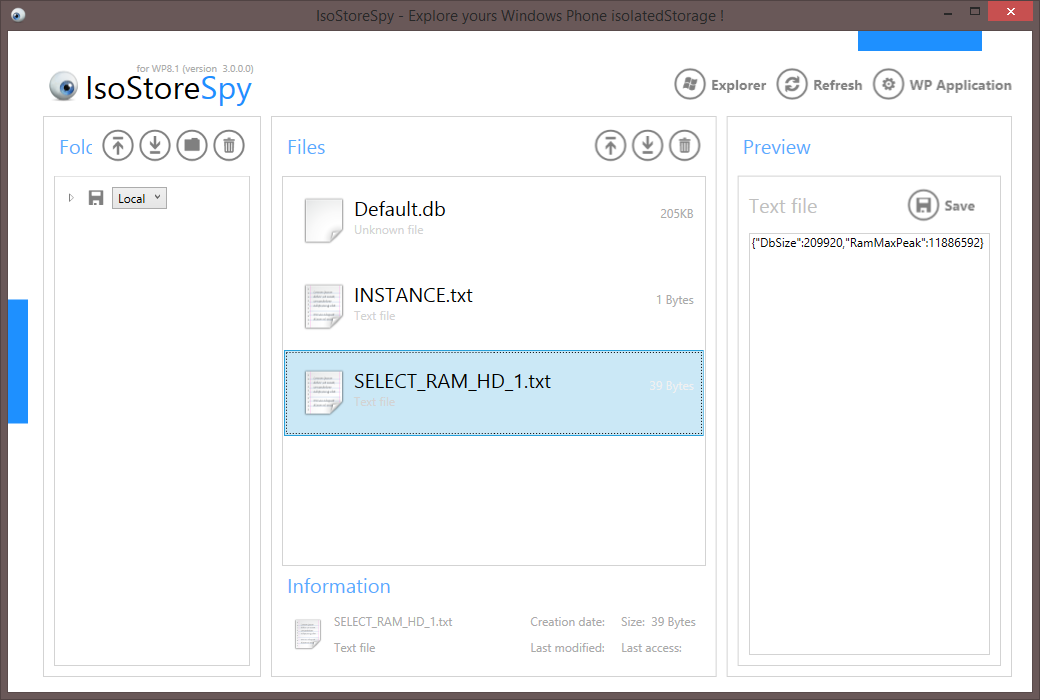
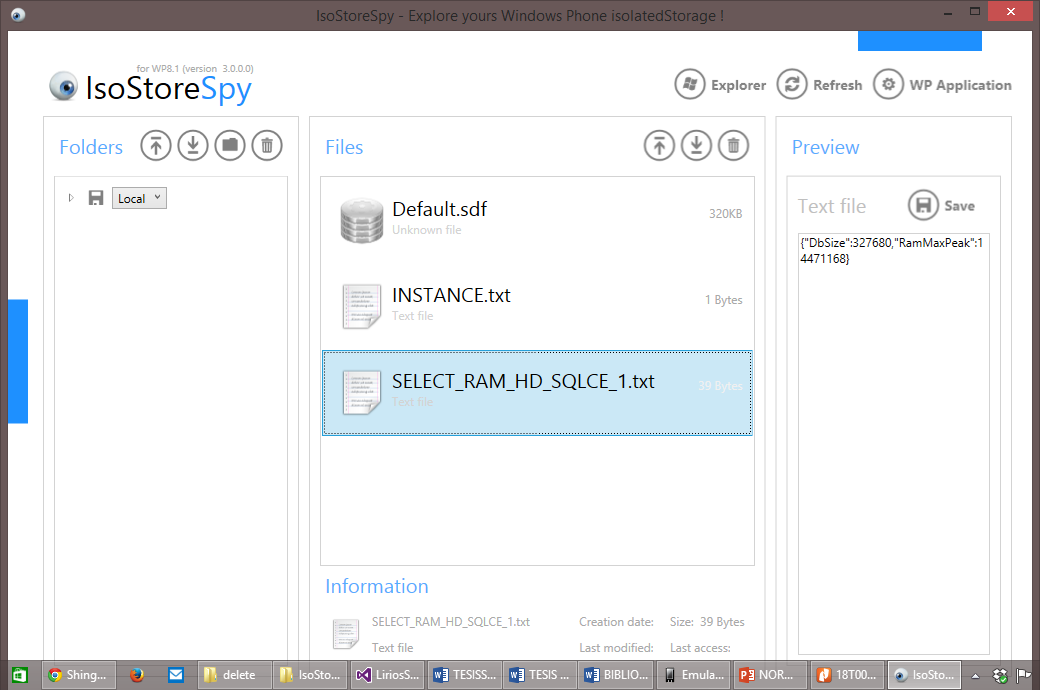
Durante el proceso de ejecución del prototipo, este realiza los siguientes pasos:

* Verificar si existe el archivo de base de datos, en caso de ser afirmativo lo elimina.
* Restaurar un nuevo archivo de BD previamente lleno de datos.
* Realiza la consulta de 1200 registros al azar.
* Capturar el pico máximo de memoria utilizado por el prototipo y el tamaño del archivo de base de datos resultante. Estos valores se los almacena en la carpeta local del almacenamiento aislado de la aplicación.
* Forzar la terminación de la ejecución de la aplicación mediante código.

Una vez que la aplicación termina su ejecución, la herramienta de profiling inicia a consolidar los datos capturados y los muestra tal como indica la figura 2-A. De este reporte se guarda el archivo de marcas que contiene el tiempo de ejecución de la aplicación y las unidades de procesamiento utilizadas por le CPU.

* **Almacenamiento aislado**

Para acceder al almacenamiento aislado del prototipo, se utiliza la herramienta IsoStoreSpy con la que se extrae el archivo de captura con los datos del pico máximo de uso memoria RAM y tamaño de archivo de base de datos resultante durante la ejecución del prototipo. Este archivo tiene por nombre “SELECT\_RAM\_HD\_SQLCE\_<número de ejecución del prototipo>.txt” para SQL Server Compact y “SELECT\_RAM\_HD\_<número de ejecución del prototipo>.txt” para SQLite como se ilustra en la figura 9-A4:



Vista del almacenamiento aislado del prototipo SELECT (SQL Server Compact, SQLite,) con la herramienta IsoStoreSpy.

**FUENTE**: Darwin Cedeño, Diego Quinatoa 2015

## ANEXO B: Tarjetas de historia de usuario.

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HT1 |
| **Nombre** | Diseño de la base de datos LiriosFadminDB |
| **Prioridad** | 5 |
| **Estimación Complejidad** | 5 |
| **Descripción** | Como administrador deseo diseñar la base de datos para almacenar la información. |
| **Tareas de ingeniería** | - Generar el diagrama de la base de datos en papel.  -Generar el Script de la base de datos. |
| **Responsable** | Equipo |
| **Tipo de Usuario** | Administrador |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HT2 |
| **Nombre** | Diseño de la arquitectura del sistema. |
| **Prioridad** | 5 |
| **Estimación Complejidad** | 3 |
| **Descripción** | Como administrador deseo diseñar la arquitectura del sistema para tener mejor visión de la solución. |
| **Tareas de ingeniería** | -Generar el diagrama de despliegue de -la solución.  -Generar el diagrama de componentes de LiriosDesktop.  -Generar el diagrama de componentes de LiriosMobile. |
| **Responsable** | Equipo |
| **Tipo de Usuario** | Administrador |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HT3 |
| **Nombre** | Instalación de herramientas |
| **Prioridad** | 5 |
| **Estimación Complejidad** | 1 |
| **Descripción** | Como administrador instalar las herramientas necesarias para el desarrollo del sistema. |
| **Tareas de ingeniería** | -Instalar Visual Studio.  -Instalar SLQServer  -Instalar Internet Information Service. |
| **Responsable** | Equipo |
| **Tipo de Usuario** | Administrador |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HU1 |
| **Nombre** | Inicio de sesión |
| **Prioridad** | 5 |
| **Estimación Complejidad** | 4 |
| **Descripción** | Como administrador deseo que exista un inicio de sesión para validar y controlar el acceso autorizado del sistema. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices.  - Generar el método de servicio web.  -Generar la función para consumir el método del servicio web.  -Interfaz gráfica. |
| **Responsable** | Darwin Cedeño |
| **Tipo de Usuario** | Administrador |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HU2 |
| **Nombre** | Registro de usuario |
| **Prioridad** | 5 |
| **Estimación Complejidad** | 3 |
| **Descripción** | Como administrador deseo registrar un usuario para controlar el acceso a los distintos módulos del sistema. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices.  - Generar el método de servicio web.  -Generar la función para consumir el método del servicio web.  -Interfaz gráfica. |
| **Responsable** | Darwin Cedeño |
| **Tipo de Usuario** | Administrador |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HU3 |
| **Nombre** | Actualizar un usuario |
| **Prioridad** | 5 |
| **Estimación Complejidad** | 3 |
| **Descripción** | Como administrador deseo actualizar los datos del usuario para tener información actual. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices para actualizar.  -Generar la función de acceso a la base de datos en LiriosWebServices para buscar.  -Generar el método de servicio web para actualizar.  - Generar el método de servicio web para buscar.  -Generar la función para consumir el método del servicio web para actualizar.  -Generar la función para consumir el método del servicio web para buscar.  -Interfaz gráfica. |
| **Responsable** | Darwin Cedeño |
| **Tipo de Usuario** | Administrador |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HU4 |
| **Nombre** | Registrar las características de habitación |
| **Prioridad** | 5 |
| **Estimación Complejidad** | 3 |
| **Descripción** | Como administrador o gerente deseo registrar las características de un tipo de habitación para saber qué clase de servicio tiene el hotel. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices.  - Generar el método de servicio web.  -Generar la función para consumir el método del servicio web.  -Interfaz gráfica. |
| **Responsable** | Darwin Cedeño |
| **Tipo de Usuario** | Administrador, Gerente |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HU5 |
| **Nombre** | Actualizar una característica de habitación. |
| **Prioridad** | 5 |
| **Estimación Complejidad** | 3 |
| **Descripción** | Como administrador o gerente deseo actualizar las características de un tipo de habitación para tener una información actual. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices para actualizar.  -Generar la función de acceso a la base de datos en LiriosWebServices para buscar.  -Generar el método de servicio web para actualizar.  - Generar el método de servicio web para buscar.  -Generar la función para consumir el método del servicio web para actualizar.  -Generar la función para consumir el método del servicio web para buscar.  -Interfaz gráfica. |
| **Responsable** | Darwin Cedeño |
| **Tipo de Usuario** | Administrador, Gerente |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HU6 |
| **Nombre** | Eliminar una característica de habitación |
| **Prioridad** | 5 |
| **Estimación Complejidad** | 3 |
| **Descripción** | Como administrador o gerente deseo eliminar las características de un tipo de habitación para tener una información actualizada. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices para eliminar.  -Generar la función de acceso a la base de datos en LiriosWebServices para buscar.  -Generar el método de servicio web para eliminar.  - Generar el método de servicio web para buscar.  -Generar la función para consumir el método del servicio web para eliminar.  -Generar la función para consumir el método del servicio web para buscar.  -Interfaz gráfica. |
| **Responsable** | Diego Quinatoa |
| **Tipo de Usuario** | Administrador, Gerente |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HU7 |
| **Nombre** | Registrar un tipo de habitación |
| **Prioridad** | 5 |
| **Estimación Complejidad** | 4 |
| **Descripción** | Como administrador o gerente deseo registrar un tipo de habitación para saber qué clase de servicio tiene el hotel. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices.  - Generar el método de servicio web.  -Generar la función para consumir el método del servicio web.  -Interfaz gráfica. |
| **Responsable** | Diego Quinatoa |
| **Tipo de Usuario** | Administrador, Gerente |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HU8 |
| **Nombre** | Actualizar un tipo de habitación |
| **Prioridad** | 5 |
| **Estimación Complejidad** | 4 |
| **Descripción** | Como administrador o gerente deseo actualizar un tipo de habitación para tener una información actual. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices para actualizar.  -Generar la función de acceso a la base de datos en LiriosWebServices para buscar.  -Generar el método de servicio web para actualizar.  - Generar el método de servicio web para buscar.  -Generar la función para consumir el método del servicio web para actualizar.  -Generar la función para consumir el método del servicio web para buscar.  -Interfaz gráfica. |
| **Responsable** | Diego Quinatoa |
| **Tipo de Usuario** | Administrador, Gerente |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HU9 |
| **Nombre** | Eliminar un tipo de habitación |
| **Prioridad** | 5 |
| **Estimación Complejidad** | 3 |
| **Descripción** | Como administrador o gerente deseo eliminar un tipo de habitación para tener una información actualizada. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices para eliminar.  -Generar la función de acceso a la base de datos en LiriosWebServices para buscar.  -Generar el método de servicio web para eliminar.  - Generar el método de servicio web para buscar.  -Generar la función para consumir el método del servicio web para eliminar.  -Generar la función para consumir el método del servicio web para buscar.  -Interfaz gráfica. |
| **Responsable** | Diego Quinatoa |
| **Tipo de Usuario** | Administrador, Gerente |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HU10 |
| **Nombre** | Registrar una habitación |
| **Prioridad** | 4 |
| **Estimación Complejidad** | 5 |
| **Descripción** | Como administrador o gerente deseo registrar una habitación para saber qué servicios tiene el hotel |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices.  - Generar el método de servicio web.  -Generar la función para consumir el método del servicio web.  -Interfaz gráfica. |
| **Responsable** | Diego Quinatoa |
| **Tipo de Usuario** | Administrador, Gerente |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HU11 |
| **Nombre** | Registrar un producto |
| **Prioridad** | 4 |
| **Estimación Complejidad** | 4 |
| **Descripción** | Como administrador o gerente deseo registrar un producto para la venta a los usuarios. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices.  - Generar el método de servicio web.  -Generar la función para consumir el método del servicio web.  -Interfaz gráfica. |
| **Responsable** | Diego Quinatoa |
| **Tipo de Usuario** | Administrador, Gerente |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HU12 |
| **Nombre** | Actualizar un producto |
| **Prioridad** | 4 |
| **Estimación Complejidad** | 5 |
| **Descripción** | Como administrador o gerente deseo actualizar un producto para tener información actualizada. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices para actualizar.  -Generar la función de acceso a la base de datos en LiriosWebServices para buscar.  -Generar el método de servicio web para actualizar.  - Generar el método de servicio web para buscar.  -Generar la función para consumir el método del servicio web para actualizar.  -Generar la función para consumir el método del servicio web para buscar.  -Interfaz gráfica. |
| **Responsable** | Diego Quinatoa |
| **Tipo de Usuario** | Administrador, Gerente |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HU13 |
| **Nombre** | Registro de un cliente |
| **Prioridad** | 4 |
| **Estimación Complejidad** | 5 |
| **Descripción** | Como recepcionista deseo registrar cliente para poder ofrecer servicios del hotel. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices.  - Generar el método de servicio web.  -Generar la función para consumir el método del servicio web.  -Interfaz gráfica. |
| **Responsable** | Darwin Cedeño |
| **Tipo de Usuario** | Recepcionista |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HU14 |
| **Nombre** | Actualizar un cliente |
| **Prioridad** | 4 |
| **Estimación Complejidad** | 5 |
| **Descripción** | Como recepcionista deseo actualizar un cliente para obtener información actual |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices para actualizar.  -Generar la función de acceso a la base de datos en LiriosWebServices para buscar.  -Generar el método de servicio web para actualizar.  - Generar el método de servicio web para buscar.  -Generar la función para consumir el método del servicio web para actualizar.  -Generar la función para consumir el método del servicio web para buscar.  -Interfaz gráfica. |
| **Responsable** | Darwin Cedeño |
| **Tipo de Usuario** | Recepcionista |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HU15 |
| **Nombre** | Crear una cuenta de servicio |
| **Prioridad** | 3 |
| **Estimación Complejidad** | 4 |
| **Descripción** | Como recepcionista deseo crear una cuenta de servicios para registrar distintos servicios del cliente. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices.  - Generar el método de servicio web.  -Generar la función para consumir el método del servicio web.  -Interfaz gráfica. |
| **Responsable** | Darwin Cedeño |
| **Tipo de Usuario** | Recepcionista |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HU16 |
| **Nombre** | Agregar un servicio de hospedaje |
| **Prioridad** | 3 |
| **Estimación Complejidad** | 4 |
| **Descripción** | Como recepcionista deseo agregar un servicio de hospedaje para asociar a una cuenta de servicio de un cliente. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices.  - Generar el método de servicio web.  -Generar la función para consumir el método del servicio web.  -Interfaz gráfica. |
| **Responsable** | Darwin Cedeño |
| **Tipo de Usuario** | Recepcionista |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HU17 |
| **Nombre** | Agregar una venta de un producto |
| **Prioridad** | 3 |
| **Estimación Complejidad** | 4 |
| **Descripción** | Como recepcionista deseo agregar una venta de un producto para asociar a una cuenta de servicio de un cliente |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices.  - Generar el método de servicio web.  -Generar la función para consumir el método del servicio web.  -Interfaz gráfica. |
| **Responsable** | Diego Quinatoa |
| **Tipo de Usuario** | Recepcionista |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HU18 |
| **Nombre** | Finalizar un servicio de hospedaje |
| **Prioridad** | 3 |
| **Estimación Complejidad** | 5 |
| **Descripción** | Como recepcionista deseo finalizar un servicio de hospedaje para cambiar el estado de una habitación de ocupado a mantenimiento y posterior uso nuevamente. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices.  - Generar el método de servicio web.  -Generar la función para consumir el método del servicio web.  -Interfaz gráfica. |
| **Responsable** | Diego Quinatoa |
| **Tipo de Usuario** | Recepcionista |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HU19 |
| **Nombre** | Cancelación de un servicio de hospedaje |
| **Prioridad** | 3 |
| **Estimación Complejidad** | 5 |
| **Descripción** | Como gerente deseo cancelar un servicio de hospedaje para no considerar como cobro por posibles errores de ingreso de datos. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices.  - Generar el método de servicio web.  -Generar la función para consumir el método del servicio web.  -Interfaz gráfica. |
| **Responsable** | Diego Quinatoa |
| **Tipo de Usuario** | Gerente |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HU20 |
| **Nombre** | Aplicar descuento del 100% |
| **Prioridad** | 3 |
| **Estimación Complejidad** | 5 |
| **Descripción** | Como gerente deseo aplicar un descuento del 100% para ofrecer hospedaje de cortesía a determinado cliente. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices.  - Generar el método de servicio web.  -Generar la función para consumir el método del servicio web.  -Interfaz gráfica. |
| **Responsable** | Diego Quinatoa |
| **Tipo de Usuario** | Gerente |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HU21 |
| **Nombre** | Agregar abonos |
| **Prioridad** | 3 |
| **Estimación Complejidad** | 3 |
| **Descripción** | Como recepcionista deseo agregar abonos para disminuir el valor a pagar en la cuenta final. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices.  - Generar el método de servicio web.  -Generar la función para consumir el método del servicio web.  -Interfaz gráfica. |
| **Responsable** | Diego Quinatoa |
| **Tipo de Usuario** | Recepcionista |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HU22 |
| **Nombre** | Finalizar cuenta de servicio |
| **Prioridad** | 3 |
| **Estimación Complejidad** | 5 |
| **Descripción** | Como Gerente, recepcionista, o administrador deseo la cuenta de servicio del cliente para proceder con el cobro de valores y su posterior facturación. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices.  - Generar el método de servicio web.  -Generar la función para consumir el método del servicio web.  -Interfaz gráfica. |
| **Responsable** | Diego Quinatoa |
| **Tipo de Usuario** | Administrador, Gerente, Recepcionista |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HU23 |
| **Nombre** | Visualizar detalle de la cuenta de servicios |
| **Prioridad** | 3 |
| **Estimación Complejidad** | 5 |
| **Descripción** | Como Gerente, recepcionista o administrador deseo visualizar el detalle de la de servicios del cliente para indicar los valores a pagar. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices.  - Generar el método de servicio web.  -Generar la función para consumir el método del servicio web.  -Interfaz gráfica. |
| **Responsable** | Diego Quinatoa |
| **Tipo de Usuario** | Administrador, Gerente, Recepcionista |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HUM1 |
| **Nombre** | Inicio de sesión Cliente |
| **Prioridad** | 2 |
| **Estimación Complejidad** | 4 |
| **Descripción** | Como cliente deseo iniciar sesión en la aplicación móvil para poder acceder a los servicios del hotel. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices.  - Generar el método de servicio web.  -Generar la función para consumir el método del servicio web.  -Interfaz gráfica. |
| **Responsable** | Diego Quinatoa |
| **Tipo de Usuario** | Cliente |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HUM2 |
| **Nombre** | Realizar pedido a la habitación |
| **Prioridad** | 2 |
| **Estimación Complejidad** | 4 |
| **Descripción** | Como cliente deseo realizar un pedido a la habitación a través de la aplicación móvil para cubrir las necesidades y deseos del cliente. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices.  - Generar el método de servicio web.  -Generar la función para consumir el método del servicio web.  -Interfaz gráfica. |
| **Responsable** | Diego Quinatoa |
| **Tipo de Usuario** | Clliente |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HUM3 |
| **Nombre** | Consulta de pedidos realizados |
| **Prioridad** | 2 |
| **Estimación Complejidad** | 5 |
| **Descripción** | Como cliente deseo listar los pedidos realizados para visualizar como cliente lo que he solicitado. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices.  - Generar el método de servicio web.  -Generar la función para consumir el método del servicio web.  -Interfaz gráfica. |
| **Responsable** | Diego Quinatoa |
| **Tipo de Usuario** | Cliente |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HUM4 |
| **Nombre** | Detalle de la cuenta de servicios |
| **Prioridad** | 2 |
| **Estimación Complejidad** | 3 |
| **Descripción** | Como cliente deseo visualizar el detalle de la cuenta de servicios para tener en cuenta lo que se debe pagar. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices.  - Generar el método de servicio web.  -Generar la función para consumir el método del servicio web.  -Interfaz gráfica. |
| **Responsable** | Diego Quinatoa |
| **Tipo de Usuario** | Cliente |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HUM5 |
| **Nombre** | Inicio de sesión camarero |
| **Prioridad** | 2 |
| **Estimación Complejidad** | 3 |
| **Descripción** | Como camarero deseo iniciar sesión en la aplicación móvil para realizar actividades en la habitación |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices.  - Generar el método de servicio web.  -Generar la función para consumir el método del servicio web.  -Interfaz gráfica. |
| **Responsable** | Diego Quinatoa |
| **Tipo de Usuario** | Camarero |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HUM6 |
| **Nombre** | Entrega de un producto |
| **Prioridad** | 2 |
| **Estimación Complejidad** | 4 |
| **Descripción** | Como camarero deseo registrar la entrega de un producto para confirmar el pedido. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices.  - Generar el método de servicio web.  -Generar la función para consumir el método del servicio web.  -Interfaz gráfica. |
| **Responsable** | Diego Quinatoa |
| **Tipo de Usuario** | Camarero |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HU24 |
| **Nombre** | Visualizar listado de habitaciones |
| **Prioridad** | 1 |
| **Estimación Complejidad** | 3 |
| **Descripción** | Como administrador, recepcionista o gerente deseo visualizar el listado de habitaciones en mantenimiento para enviar a una camarera que realiza su respectivo trabajo. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices.  - Generar el método de servicio web.  -Generar la función para consumir el método del servicio web.  -Interfaz gráfica. |
| **Responsable** | Diego Quinatoa |
| **Tipo de Usuario** | Administrador, Gerente, Recepcionista |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HU25 |
| **Nombre** | Actualizar el estado de una habitación |
| **Prioridad** | 1 |
| **Estimación Complejidad** | 3 |
| **Descripción** | Como administrador, recepcionista o gerente deseo actualizar el estado de una habitación a disponible para incluirla de nuevo en los servicios del hotel. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices.  - Generar el método de servicio web.  -Generar la función para consumir el método del servicio web.  -Interfaz gráfica. |
| **Responsable** | Diego Quinatoa |
| **Tipo de Usuario** | Administrador, Gerente, Recepcionista |

|  |  |
| --- | --- |
| **HISTORIA DE USUARIO** | |
| **Id** | HU26 |
| **Nombre** | Notificaciones |
| **Prioridad** | 1 |
| **Estimación Complejidad** | 3 |
| **Descripción** | Como recepcionista deseo visualizar notificaciones referente a las habitaciones en mantenimiento, como pedidos a la habitación. |
| **Tareas de ingeniería** | -Generar la función de acceso a la base de datos en LiriosWebServices.  - Generar el método de servicio web.  -Generar la función para consumir el método del servicio web.  -Interfaz gráfica. |
| **Responsable** | Diego Quinatoa |
| **Tipo de Usuario** | Recepcionista |

## ANEXO C: Pruebas de aceptación.

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA1 |
| **Historia a la que pertenece** | HT1 |
| **Descripción** | Diseño de la Base de Datos LiriosFadminDB |
| **Precondiciones** | ninguna |
| **Ejecución** | 1.- Diseñar tablas.  2.- Generar el Script de la base de datos con tablas, funciones y data preliminar. |
| **Resultado** | Base de datos generada. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA2 |
| **Historia a la que pertenece** | HT2 |
| **Descripción** | Diseño de la arquitectura del sistema. |
| **Precondiciones** | ninguna |
| **Ejecución** | 1.- Generar el diagrama de componentes. |
| **Resultado** | Diagrama generado. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA3 |
| **Historia a la que pertenece** | HT3 |
| **Descripción** | Instalación de herramientas |
| **Precondiciones** | ninguna |
| **Ejecución** | 1.- Instalar las herramientas y componentes necesarios para el desarrollo del sistema. |
| **Resultado** | Instalación realizada |
| **Responsable** | Diego Quinatoa |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA4 |
| **Historia a la que pertenece** | HU1 |
| **Descripción** | Inicio de sesión |
| **Precondiciones** | - Poseer una conexión a internet  - El usuario no debe tener sesiones activas.  - El usuario debe estar ubicado en la pantalla de login. |
| **Ejecución** | 1.- Se ingresa el usuario que es el número de cédula.  2.- Se ingresa la contraseña.  3.- Se hace click en el botón iniciar sesión. |
| **Resultado** | Mensaje de error en el caso de que los datos no sean correctos. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA5 |
| **Historia a la que pertenece** | HU1 |
| **Descripción** | Inicio de sesión |
| **Precondiciones** | - Poseer una conexión a internet  - El usuario no debe tener sesiones activas.  - El usuario debe estar ubicado en la pantalla de login. |
| **Ejecución** | 1.- Se ingresa el usuario que es el número de cédula.  2.- Se ingresa la contraseña.  3.- Se hace click en el botón iniciar sesión. |
| **Resultado** | Se muestra el menú principal |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA6 |
| **Historia a la que pertenece** | HU2 |
| **Descripción** | Registro de Usuario |
| **Precondiciones** | - Haber iniciado sesión con rol administrador  - Conexión internet |
| **Ejecución** | 1.- Click en la opción gestionar usuario  2.- Ingresar los datos del usuario.  3.- Dar click en el botón guardar. |
| **Resultado** | Mensaje en pantalla usuario guardado exitosamente. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA7 |
| **Historia a la que pertenece** | HU2 |
| **Descripción** | Registro de Usuario |
| **Precondiciones** | - Haber iniciado sesión con rol administrador  - Conexión internet |
| **Ejecución** | 1.- Click en la opción gestionar usuario  2.- Ingresar los datos del usuario.  3.- Dar click en el botón guardar. |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA8 |
| **Historia a la que pertenece** | HU3 |
| **Descripción** | Actualizar un Usuario |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet |
| **Ejecución** | 1.- Click en la opción gestionar usuario  2.- Ingresar en el cuadro de búsqueda del usuario la cedula, nombres o apellidos.  3.- Dar click en el elemento encontrado.  4.- Ingresar los datos a actualizar.  5.- Click en el botón actualizar |
| **Resultado** | Mensaje en pantalla los datos se actualizaron exitosamente |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA9 |
| **Historia a la que pertenece** | HU3 |
| **Descripción** | Actualizar un Usuario |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet |
| **Ejecución** | 1.- Click en la opción gestionar usuario  2.- Ingresar en el cuadro de búsqueda del usuario la cedula, nombres o apellidos.  3.- Dar click en el elemento encontrado.  4.- Ingresar los datos a actualizar.  5.- Click en el botón actualizar |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA10 |
| **Historia a la que pertenece** | HU4 |
| **Descripción** | Registrar las características de habitación |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet |
| **Ejecución** | 1.- Click en la opción gestionar característica de habitación.  2.- Ingresar los datos de la característica de la habitación.  3.- Dar click en el botón guardar. |
| **Resultado** | Mensaje en pantalla los datos se guardaron exitosamente. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA11 |
| **Historia a la que pertenece** | HU4 |
| **Descripción** | Registrar las características de habitación |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet |
| **Ejecución** | 1.- Click en la opción gestionar característica de habitación.  2.- Ingresar los datos de la característica de la habitación.  3.- Dar click en el botón guardar. |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA12 |
| **Historia a la que pertenece** | HU5 |
| **Descripción** | Actualizar una característica de habitación |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet |
| **Ejecución** | 1.- Click en la opción gestionar característica de habitación.  2.- Seleccionar la característica habitación a actualizar.  3.- Ingresar los datos a actualizar.  4.- Click en el botón actualizar |
| **Resultado** | Mensaje en pantalla los datos se actualizaron exitosamente. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA13 |
| **Historia a la que pertenece** | HU5 |
| **Descripción** | Actualizar una característica de habitación |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet |
| **Ejecución** | 1.- Click en la opción gestionar característica de habitación.  2.- Seleccionar la característica habitación a actualizar.  3.- Ingresar los datos a actualizar.  4.- Click en el botón actualizar |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA14 |
| **Historia a la que pertenece** | HU6 |
| **Descripción** | Eliminar una característica de habitación |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet |
| **Ejecución** | 1.- Click en la opción gestionar característica de habitación.  2.- Seleccionar la característica habitación a eliminar.  3.- Click en el botón eliminar. |
| **Resultado** | Mensaje en pantalla los datos se eliminaron exitosamente. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA15 |
| **Historia a la que pertenece** | HU6 |
| **Descripción** | Eliminar una característica de habitación |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet |
| **Ejecución** | 1.- Click en la opción gestionar característica de habitación.  2.- Seleccionar la característica habitación a eliminar.  3.- Click en el botón eliminar. |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA16 |
| **Historia a la que pertenece** | HU7 |
| **Descripción** | Registrar un tipo de habitación |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet |
| **Ejecución** | 1.- Click en la opción gestionar tipo de habitación.  2.- Ingresar los datos del tipo de habitación.  3.- Seleccionar las características para el tipo de habitación.  3.- Dar click en el botón guardar. |
| **Resultado** | Mensaje en pantalla los datos se guardaron exitosamente. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA17 |
| **Historia a la que pertenece** | HU7 |
| **Descripción** | Registrar un tipo de habitación |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet |
| **Ejecución** | 1.- Click en la opción gestionar tipo de habitación.  2.- Ingresar los datos del tipo de habitación.  3.- Seleccionar las características para el tipo de habitación.  3.- Dar click en el botón guardar. |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA18 |
| **Historia a la que pertenece** | HU8 |
| **Descripción** | Actualizar un tipo de habitación |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet |
| **Ejecución** | 1.- Click en la opción gestionar tipo de habitación.  2.- Seleccionar un tipo de habitación.  3.- Ingresar el dato a modificar.  4.- Dar click en el botón guardar. |
| **Resultado** | Mensaje en pantalla los datos se actualizaron exitosamente. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA19 |
| **Historia a la que pertenece** | HU8 |
| **Descripción** | Actualizar un tipo de habitación |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet |
| **Ejecución** | 1.- Click en la opción gestionar tipo de habitación.  2.- Seleccionar un tipo de habitación.  3.- Ingresar el dato a modificar.  4.- Dar click en el botón guardar. |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA20 |
| **Historia a la que pertenece** | HU9 |
| **Descripción** | Eliminar un tipo de habitación |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet |
| **Ejecución** | 1.- Click en la opción gestionar tipo de habitación.  2.- Seleccionar un tipo de habitación.  3.- Dar click en el botón eliminar. |
| **Resultado** | Mensaje en pantalla los datos se eliminaron exitosamente. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA21 |
| **Historia a la que pertenece** | HU9 |
| **Descripción** | Eliminar un tipo de habitación |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet |
| **Ejecución** | 1.- Click en la opción gestionar tipo de habitación.  2.- Seleccionar un tipo de habitación.  3.- Dar click en el botón eliminar. |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA22 |
| **Historia a la que pertenece** | HU10 |
| **Descripción** | Registrar una habitación |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet |
| **Ejecución** | 1.- Click en gestionar habitación.  2.- Ingresar los datos de la habitación.  3.- Click en el botón guardar. |
| **Resultado** | Mensaje en pantalla los datos se guardaron exitosamente. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA23 |
| **Historia a la que pertenece** | HU10 |
| **Descripción** | Registrar una habitación |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet |
| **Ejecución** | 1.- Click en gestionar habitación.  2.- Ingresar los datos de la habitación.  3.- Click en el botón guardar. |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA24 |
| **Historia a la que pertenece** | HU11 |
| **Descripción** | Registrar un producto |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet |
| **Ejecución** | 1.- Click en gestionar productos.  2.- Ingresar los datos del producto.  3.- Click en el botón guardar. |
| **Resultado** | Mensaje en pantalla los datos se guardaron exitosamente. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA25 |
| **Historia a la que pertenece** | HU11 |
| **Descripción** | Registrar un producto |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet |
| **Ejecución** | 1.- Click en gestionar productos.  2.- Ingresar los datos del producto.  3.- Click en el botón guardar. |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA26 |
| **Historia a la que pertenece** | HU12 |
| **Descripción** | Actualizar un producto |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet |
| **Ejecución** | 1.- Click en gestionar productos.  2.- Buscar el producto mediante código de barras, categoría o descripción.  3.- Seleccionar el producto  4.- Ingreso el dato actualizar  3.- Click en el botón actualizar. |
| **Resultado** | Mensaje en pantalla los datos se actualizaron exitosamente. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA27 |
| **Historia a la que pertenece** | HU12 |
| **Descripción** | Actualizar un producto |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet |
| **Ejecución** | 1.- Click en gestionar productos.  2.- Buscar el producto mediante código de barras, categoría o descripción.  3.- Seleccionar el producto  4.- Ingreso el dato actualizar  3.- Click en el botón actualizar. |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA28 |
| **Historia a la que pertenece** | HU13 |
| **Descripción** | Registro de un cliente |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet |
| **Ejecución** | 1.- Click en gestionar cliente.  2.- Ingresar los datos del cliente.  3.- Click en el botón guardar. |
| **Resultado** | Mensaje en pantalla los datos se actualizaron exitosamente. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA29 |
| **Historia a la que pertenece** | HU13 |
| **Descripción** | Registro de un cliente |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet |
| **Ejecución** | 1.- Click en gestionar cliente.  2.- Ingresar los datos del cliente.  3.- Click en el botón guardar. |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA30 |
| **Historia a la que pertenece** | HU14 |
| **Descripción** | Actualizar un cliente |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet |
| **Ejecución** | 1.- Click en gestionar cliente.  2.- Buscar el cliente mediante cedula, nombres o apellidos.  3.- Seleccionar el cliente.  4.- Ingreso el dato actualizar  3.- Click en el botón actualizar. |
| **Resultado** | Mensaje en pantalla los datos se actualizaron exitosamente. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA31 |
| **Historia a la que pertenece** | HU14 |
| **Descripción** | Actualizar un cliente |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet |
| **Ejecución** | 1.- Click en gestionar cliente.  2.- Buscar el cliente mediante cedula, nombres o apellidos.  3.- Seleccionar el cliente.  4.- Ingreso el dato actualizar  3.- Click en el botón actualizar. |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA32 |
| **Historia a la que pertenece** | HU15 |
| **Descripción** | Crear una cuenta de servicio |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet |
| **Ejecución** | 1.-Click en opción ventas  2.- Buscar cliente por cedula.  3.- Click en botón crear cuentas de servicio. |
| **Resultado** | Mensaje en pantalla se creó exitosamente. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA33 |
| **Historia a la que pertenece** | HU15 |
| **Descripción** | Crear una cuenta de servicio |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet |
| **Ejecución** | 1.-Click en opción ventas  2.- Buscar cliente por cedula.  3.- Click en botón crear cuentas de servicio. |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA34 |
| **Historia a la que pertenece** | HU16 |
| **Descripción** | Agregar un servicio de hospedaje |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet  - Estar en ubicado en la pantalla de ventas. |
| **Ejecución** | 1.- Click en la opción agregar servicio de hospedaje.  2.- Escoger la habitación.  3. Seleccionó el número de adultos y de niños.  4.- Click en el botón agregar. |
| **Resultado** | Mensaje en pantalla se agregó un servicio de hospedaje exitosamente. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA35 |
| **Historia a la que pertenece** | HU16 |
| **Descripción** | Agregar un servicio de hospedaje |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet  - Estar en ubicado en la pantalla de ventas. |
| **Ejecución** | 1.- Click en la opción agregar servicio de hospedaje.  2.- Escoger la habitación.  3. Seleccionó el número de adultos y de niños.  4.- Click en el botón agregar. |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA36 |
| **Historia a la que pertenece** | HU17 |
| **Descripción** | Agregar una venta de un producto |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet  - Estar en ubicado en la pantalla de ventas. |
| **Ejecución** | 1.- Click en la opción venta de productos.  2.- Buscar un producto.  3.- seleccionar el producto  4.- Ingresar la cantidad de producto a vender. |
| **Resultado** | Mensaje en pantalla se agregó una venta de producto exitosamente. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA37 |
| **Historia a la que pertenece** | HU17 |
| **Descripción** | Agregar una venta de un producto |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet  - Estar en ubicado en la pantalla de ventas. |
| **Ejecución** | 1.- Click en la opción venta de productos.  2.- Buscar un producto.  3.- seleccionar el producto  4.- Ingresar la cantidad de producto a vender. |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA38 |
| **Historia a la que pertenece** | HU18 |
| **Descripción** | Finalizar un servicio de hospedaje |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet  - Estar en ubicado en la pantalla de ventas. |
| **Ejecución** | 1.- Click derecho en uno de los servicios de hospedaje.  2.- Click en la opción finalizar. |
| **Resultado** | Finalización de hospedaje exitosa. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA39 |
| **Historia a la que pertenece** | HU18 |
| **Descripción** | Finalizar un servicio de hospedaje |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet  - Estar en ubicado en la pantalla de ventas. |
| **Ejecución** | 1.- Click derecho en uno de los servicios de hospedaje.  2.- Click en la opción finalizar. |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA40 |
| **Historia a la que pertenece** | HU19 |
| **Descripción** | Cancelación de un servicio de hospedaje |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet  - Estar en ubicado en la pantalla de ventas. |
| **Ejecución** | 1.- Click derecho en uno de los servicios de hospedaje.  2.- Click en la opción cancelar. |
| **Resultado** | Cancelar un servicio de hospedaje exitosamente |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA41 |
| **Historia a la que pertenece** | HU19 |
| **Descripción** | Cancelación de un servicio de hospedaje |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet  - Estar en ubicado en la pantalla de ventas. |
| **Ejecución** | 1.- Click derecho en uno de los servicios de hospedaje.  2.- Click en la opción cancelar. |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA42 |
| **Historia a la que pertenece** | HU20 |
| **Descripción** | Aplicar descuento del 100% |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet  - Estar en ubicado en la pantalla de ventas. |
| **Ejecución** | 1.- Click derecho en uno de los servicios de hospedaje.  2.- Click en la opción aplicar 100 %. |
| **Resultado** | Descuento aplicado correctamente. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA43 |
| **Historia a la que pertenece** | HU20 |
| **Descripción** | Aplicar descuento del 100% |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet  - Estar en ubicado en la pantalla de ventas. |
| **Ejecución** | 1.- Click derecho en uno de los servicios de hospedaje.  2.- Click en la opción aplicar 100 %. |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA44 |
| **Historia a la que pertenece** | HU21 |
| **Descripción** | Agregar abonos |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet  - Estar en ubicado en la pantalla de ventas. |
| **Ejecución** | 1.- Ingresar el valor abonar.  2.- Click en el botón abonar. |
| **Resultado** | Valor abonado exitosamente |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA45 |
| **Historia a la que pertenece** | HU21 |
| **Descripción** | Agregar abonos |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet  - Estar en ubicado en la pantalla de ventas. |
| **Ejecución** | 1.- Ingresar el valor abonar.  2.- Click en el botón abonar. |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA46 |
| **Historia a la que pertenece** | HU22 |
| **Descripción** | Finalizar cuenta de servicio |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet  - Estar en ubicado en la pantalla de ventas. |
| **Ejecución** | 1.- Click en el botón finalizar cuenta de servicio. |
| **Resultado** | Servicio finalizado correctamente |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA47 |
| **Historia a la que pertenece** | HU22 |
| **Descripción** | Finalizar cuenta de servicio |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet  - Estar en ubicado en la pantalla de ventas. |
| **Ejecución** | 1.- Click en el botón finalizar cuenta de servicio. |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA48 |
| **Historia a la que pertenece** | HU23 |
| **Descripción** | Visualizar detalle de la cuenta de servicios |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet  - Estar en ubicado en la pantalla de ventas. |
| **Ejecución** | 1.- Click en el botón actualizar |
| **Resultado** | Visualización correcta. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA49 |
| **Historia a la que pertenece** | HU23 |
| **Descripción** | Visualizar detalle de la cuenta de servicios |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet  - Estar en ubicado en la pantalla de ventas. |
| **Ejecución** | 1.- Click en el botón actualizar |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA50 |
| **Historia a la que pertenece** | HU24 |
| **Descripción** | Visualizar el listado de habitaciones |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet. |
| **Ejecución** | 1.- Click en la opción visualizar habitaciones en mantenimiento. |
| **Resultado** | Visualización correcta. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA51 |
| **Historia a la que pertenece** | HU24 |
| **Descripción** | Visualizar el listado de habitaciones |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet. |
| **Ejecución** | 1.- Click en la opción visualizar habitaciones en mantenimiento. |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA52 |
| **Historia a la que pertenece** | HU25 |
| **Descripción** | Actualizar el estado de una habitación |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet  - Estar en ubicado en la pantalla de listado de pantalla en mantenimiento. |
| **Ejecución** | 1.- Click en el botón cambiar al estado disponible. |
| **Resultado** | Actualizado del estado de una habitación correctamente. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA53 |
| **Historia a la que pertenece** | HU25 |
| **Descripción** | Actualizar el estado de una habitación |
| **Precondiciones** | - Haber iniciado sesión con el rol administrador  -Conexión internet  - Estar en ubicado en la pantalla de listado de pantalla en mantenimiento. |
| **Ejecución** | 1.- Click en el botón cambiar al estado disponible. |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA54 |
| **Historia a la que pertenece** | HU26 |
| **Descripción** | Notificaciones recepcionista |
| **Precondiciones** | -Conexión internet |
| **Ejecución** | ninguna |
| **Resultado** | Mostrar una notificación en el sistema operativo |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA55 |
| **Historia a la que pertenece** | HU26 |
| **Descripción** | Notificaciones recepcionista |
| **Precondiciones** | -Conexión internet |
| **Ejecución** | ninguna |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA56 |
| **Historia a la que pertenece** | HUM1 |
| **Descripción** | Inicio de sesión Cliente |
| **Precondiciones** | -Conexión internet |
| **Ejecución** | 1.- Ingresar la cedula y la clave móvil.  2.- Click en el botón iniciar sesión. |
| **Resultado** | Pantalla de menú principal. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA57 |
| **Historia a la que pertenece** | HUM1 |
| **Descripción** | Inicio de sesión Cliente |
| **Precondiciones** | -Conexión internet |
| **Ejecución** | 1.- Ingresar la cedula y la clave móvil.  2.- Click en el botón iniciar sesión. |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA58 |
| **Historia a la que pertenece** | HUM2 |
| **Descripción** | Realizar pedido a la habitación |
| **Precondiciones** | - Haber iniciado sesión  -Conexión internet  - Estar en ubicado en el menú principal. |
| **Ejecución** | 1.- Click en el botón realizar pedido a la habitación.  2.- Buscar el producto.  3.- Ingresar la cantidad.  4.- Click en el botón solicitar. |
| **Resultado** | Se realizó el pedido correctamente |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA59 |
| **Historia a la que pertenece** | HUM2 |
| **Descripción** | Realizar pedido a la habitación |
| **Precondiciones** | - Haber iniciado sesión  -Conexión internet  - Estar en ubicado en el menú principal. |
| **Ejecución** | 1.- Click en el botón realizar pedido a la habitación.  2.- Buscar el producto.  3.- Ingresar la cantidad.  4.- Click en el botón solicitar. |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA60 |
| **Historia a la que pertenece** | HUM3 |
| **Descripción** | Consulta de pedidos realizados |
| **Precondiciones** | - Haber iniciado sesión  -Conexión internet |
| **Ejecución** | 1.- Click en el botón ver pedidos realizados |
| **Resultado** | Visualizados correctamente |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA61 |
| **Historia a la que pertenece** | HUM3 |
| **Descripción** | Consulta de pedidos realizados |
| **Precondiciones** | - Haber iniciado sesión  -Conexión internet |
| **Ejecución** | 1.- Click en el botón ver pedidos realizados |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA62 |
| **Historia a la que pertenece** | HUM4 |
| **Descripción** | Detalle de la cuenta de servicios |
| **Precondiciones** | - Haber iniciado sesión  -Conexión internet |
| **Ejecución** | 1.- Click en el botón detalle cuenta. |
| **Resultado** | Visualizar correctamente |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA63 |
| **Historia a la que pertenece** | HUM4 |
| **Descripción** | Detalle de la cuenta de servicios |
| **Precondiciones** | - Haber iniciado sesión  -Conexión internet |
| **Ejecución** | 1.- Click en el botón detalle cuenta. |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA64 |
| **Historia a la que pertenece** | HUM5 |
| **Descripción** | Inicio de sesión camarero |
| **Precondiciones** | -Conexión internet |
| **Ejecución** | 1.- Ingresar el usuario y contraseña  2.- Click en el botón iniciar sesión |
| **Resultado** | Pantalla de menú principal. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA65 |
| **Historia a la que pertenece** | HUM5 |
| **Descripción** | Inicio de sesión camarero |
| **Precondiciones** | -Conexión internet |
| **Ejecución** | 1.- Ingresar el usuario y contraseña  2.- Click en el botón iniciar sesión |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA66 |
| **Historia a la que pertenece** | HUM6 |
| **Descripción** | Entrega de un producto |
| **Precondiciones** | - Haber iniciado sesión  -Conexión internet |
| **Ejecución** | 1.- Click en la opción entrega de productos  2.- seleccionar el producto a entregar  3.- Ingresar el código de orden proporcionado por el cliente.  4.- Click en el botón confirmar |
| **Resultado** | Entrega confirmada |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Id** | PA67 |
| **Historia a la que pertenece** | HUM6 |
| **Descripción** | Entrega de un producto |
| **Precondiciones** | - Haber iniciado sesión  -Conexión internet |
| **Ejecución** | 1.- Click en la opción entrega de productos  2.- seleccionar el producto a entregar  3.- Ingresar el código de orden proporcionado por el cliente.  4.- Click en el botón confirmar |
| **Resultado** | Mensaje en pantalla definido en la clase de manejo de mensaje de error. |
| **Responsable** | Darwin Cedeño |
| **Evaluación** | Exitosa. |