Universidad Autónoma del Estado de México

Centro Universitario UAEM Zumpango

Ingeniería en Computación

Tesis

"SISTEMA DE INFORMACIÓN PARA GESTIÓN DE ALUMNOS DE INGENIERÍA EN COMPUTACIÓN DEL CU UAEM ZUMPANGO"

Que para obtener el Título de

Ingeniera en Computación

Presenta:

Ana Karen Pérez García

Asesor:

Dr. en C. C. Asdrúbal López Chau

Resumen

Las bases de datos (BD) son sistemas de información que junto con los gestores de base da datos almacenan y manipulan información. El propósito de esta tesis es resolver el problema de almacenamiento de los datos de la Ingeniería en computación del centro universitario UAEM Zumpango, así como también dar a conocer las características, uso, diseño de las BD. Una BD tiene mucha importancia ya que la información que se maneja actualmente es más compleja de almacenar en archivos.

Los gestores de base de datos (SGBD) son los encargados de manejar la información en programas, así como también la elección de la estructura para el almacenamiento y búsqueda de los datos, además el SGBD facilita al usuario el uso, la descripción y manejo de los datos. Esta herramienta es efectiva, porque permite el acceso a varios usuarios al mismo tiempo.

MySQL es un gestor de BD relacionales de uso rápido y flexible de código abierto ya que cuenta con licencia dual GPL/Licencia comercial, ideal para aplicaciones web y trabaja en distinto sistemas operativos como Windows, GNU/Linux, FreeBSD, Mac OS X, por mencionar algunos. Sus características principales de las versiones recientes son: las extensiones del lenguaje SQL las tiene MySQL, transacciones y llaves foráneas, conectividad segura, replicación. Las ventajas que tiene este gestor son buenas, porque cumple con necesidades que lo diseñadores de BD buscar para realizar su modelo relacional.

Java es un lenguaje de programación orientada a objetos el cual esta diseñado para ser portable, es decir es un lenguaje multiplataforma. Es sencillo , portable, seguro, robusto multihilos, de alto rendimiento, dinámico y distribuido. Java tiene la sintaxis de la programación en C y C++ por el enfoque orientado a objetos.

Para concluir, este proyecto de tesis tiene una aplicación en Java que en conjunto con las BD, el gestor MySQL fue posible crear un sistema de información que organice y controle la información de los alumnos de Ingeniería en computación.

Abstract

Databases (BDs) are information systems that, together with the database managers, store and manipulate information. The purpose of this thesis is to solve the problem of storage of data of the Computer Engineering of UAEM Zumpango university center, as well as to publicize the characteristics, use, design of the BD. A BD is very important since the information that is handled at present is more complex to store in files.

Database managers (DBMS) are responsible for managing the information in programs, as well as the choice of the structure for the storage and search of data, in addition the DBMS facilitates to the user the use, description and management of the data. This tool is effective because it allows access to several users at the same time.

MySQL is a fast and flexible open source relational database manager with a dual license GPL / Business License, ideal for web applications and works on different operating systems such as Windows, GNU / Linux, FreeBSD, Mac OS X, Mention some. Its main features of recent versions are: SQL language extensions have MySQL, foreign keys and transactions, secure connectivity, replication. The advantages that this manager has are good, because it meets needs that the designers of BD look for to realize their relational model.

Java is an object-oriented programming language which is designed to be portable, that is, it is a multiplatform language. It is simple, portable, secure, robust multi-threaded, high-performance, dynamic and distributed. Java has the programming syntax in C and C ++ for the object-oriented approach.

In conclusion, this thesis project has an application in Java that together with the BD, the MySQL manager was possible to create an information system that organizes and controls the information of the students of Computer Engineering. Google Traductor para empresas:Translator ToolkitTraductor de sitios webGlobal Market Fin

Índice general

1. Introducción **12**

1.1. Planteamiento del problema . . . 13
1.2. Motivación . . . 13
1.3. Hipótesis . . . 14
1.4. Justificación . . . 14
1.5. Objetivos generales y específicos . . . 14
1.5.1. General . . . 14
1.5.2. Específicos . . . 14
1.6. Alcance . . . 15

2. Tecnologías utilizadas para el desarrollo del sistema **16**

2.1. Historia de las bases de datos . . . 16
2.2. Bases de datos . . . 18
2.2.1. Tipos de bases de datos . . . 19
2.2.2. Modelos de bases de datos . . . 20
2.2.2.1. Base de datos jerárquica . . . 20
2.2.2.2. Base de datos en red . . . 22
2.2.2.3. Base de datos orientada a objetos . . . 23
2.2.2.4. Base de datos operacionales o transaccionales . . . 24
2.2.2.5. Bases de datos relacionales . . . 25
2.3. Arquitectura de los sistemas de bases de datos relacionales . . . 26
2.3.1. Independencia de datos . . . 27
2.4. Administración de BD . . . 27
2.4.1. Diseño de base de datos . . . 28
2.4.2. Desarrollo de base de datos . . . 29
2.4.3. Gestión de Base de Datos . . . 30
2.5. Modelo relacional . . . 30
2.5.1. Entidades . . . 30
2.5.2. Interrelación . . . 31
2.5.2.1. Atributos . . . 32
2.5.2.2. Dominio . . . 32
2.5.2.3. Tabla o esquemas . . . 32

2.5.2.4. Clave primaria . . . 32
2.5.2.5. Cardinalidad . . . 33
2.5.2.6. Relación uno a muchos . . . 33
2.5.2.7. Relación uno a uno . . . 33
2.5.2.8. Relación muchos a muchos . . . 33
2.5.3. Diagrama Entidad - Relación . . . 34
2.5.3.1. Conjunto de entidades débiles . . . 35
2.6. Álgebra relacional . . . 35
2.6.1. Operaciones relacionales básicas . . . 36
2.6.1.1. SELECT . . . 36
2.6.1.2. PROJECT . . . 36
2.6.1.3. PRODUCT . . . 37
2.6.1.4. JOIN (o reunión natural) . . . 38
2.6.1.5. UNIÓN . . . 40
2.6.1.6. INTERSECT . . . 40
2.6.1.7. DIFFERENCE . . . 41
2.7. Dependencias funcionales . . . 41
2.7.1. Superclaves, claves candidatas y claves primarias . . . 41
2.8. Normalización de bases de datos . . . 42
2.8.1. Primera Forma Normal (1FN) . . . 42
2.8.2. Segunda Forma Normal (2FN) . . . 44
2.8.3. Tercera Forma Normal (3FN) . . . 44
2.9. Lenguaje estructurado de consulta SQL . . . 44
2.9.1. Consultas estructurales . . . 45
2.9.2. Tipos de datos . . . 46
2.10. Sistema Gestores de Bases de Datos (SGBD) . . . 52
2.10.1. Lenguajes de los SGBD . . . 54
2.10.1.1. Lenguajes de los SGBD . . . 54
2.10.1.2. Diccionario de datos . . . 54
2.10.1.3. Seguridad e integridad de los datos . . . 54
2.11. Java . . . 55
2.11.1. Características del lenguaje Java . . . 55
2.11.2. Java swing . . . 56

3. Desarrollo del sistema **57**
3.1. Modelo Entidad- Relacional . . . 58
3.2. Modelo Relacional . . . 60
3.3. Descripción de tablas . . . 62
3.4. Relaciones entre tablas . . . 67
3.4.1. Relación Alumno - Historial . . . 68
3.4.2. Relación Alumno - Curso . . . 69
3.4.3. Relación UA - Curso . . . 70

3.4.4. Relación UA - Historial . . . 71
3.4.5. Relación Curso - Calificación . . . 72
3.5. Nivel lógico . . . 73
3.6. Instalación MySQL Workbench . . . 82
3.7. Instalación y configuración del servidor MySQL . . . 85
3.8. Sentencias SQL . . . 94
3.9. Consultas . . . 98
3.9.1. Buscar alumnos por número de cuenta . . . 102
3.9.2. Buscar alumno por Apellido Paterno . . . 103
3.9.3. Buscar alumno por Apellido Materno . . . 104
3.9.4. Buscar alumno por Nombre . . . 105
3.9.5. Buscar alumno por periodo . . . 106
3.9.6. Buscar alumno por UA . . . 107
3.9.7. Buscar alumno por Grupo . . . 108
3.9.8. Buscar alumno por Turno . . . 109
3.9.9. Buscar alumnos por año de ingreso . . . 110
3.9.10. Calcular promedio de un alumno . . . 111
3.9.11. Calcular promedio por periodo . . . 112
3.9.12. Calcular promedio por UA . . . 113
3.9.13. Estado académico de los alumnos . . . 114
3.9.14. Alumnos inscritos por periodo . . . 115
3.9.15. Alumnos inscritos por UA . . . 116
3.9.16. Alumnos inscritos por Línea de acentuación . . . 117
3.9.17. Calificaciones por Número de Cuenta . . . 118

4. Pruebas y resultados realizadas al sistema 119
4.1. Pruebas . . . 119
4.2. Resultados . . . 120

5. Conclusiones 125

Bibliografía 126

Índice de figuras

2.1. Herman Hollerith y máquina tabuladora 17
2.2. Modelo Jerárquico 20
2.3. Modelo de base de datos en red [9] 22
2.4. Niveles ae Arquitectura de base de datos 26
2.5. Administración de BD 28
2.6. Representación de entidades 31
2.7. Ejemplo de interrelación 31
2.8. Representación de clave primaria 33
2.9. Ejemplo de un Diagrama E- R 35
2.10. Ejemplo de sentencia SELECT 36
2.11. Ejemplo de sentencia PROJECT 37
2.12. Ejemplo de sentencia PRODUCT 37
2.13. Ejemplo de sentencia JOIN 38
2.14. Ejemplo de sentencia Paso 1 PRODUCT 38
2.15. Ejemplo de sentencia Paso 2 SELECT 39
2.16. Ejemplo de sentencia Paso 3 PROJECT 39
2.17. Ejemplo de sentencia UNION 40
2.18. Ejemplo de sentencia INTERSECT 40
2.19. Ejemplo de sentencia DIFFERENCE 41
2.20. Datos generales sin normalizar 43
2.21. Ordenado a Primera Forma Normal 44

3.1. Modelo entidad-relación del sistema de desarrollo 59
3.2. Modelo relacional del sistema de desarrollado 61
3.3. Tabla Alumno. 62
3.4. Tabla UA 63
3.5. Tabla Historial 65
3.6. Tabla Calificación 66
3.7. Tabla Curso 67
3.8. Relación Alumno - Historial 68
3.9. Relación Alumno y Curso 69
3.10. Relación UA y Curso 70

3.11. Relación UA e Historial . . . 71
3.12. Relación Curso y Calificación . . . 72
3.13. Asistente de instalación de MySQL Workbench . . . 82
3.14. Localización de la instalación . . . 83
3.15. Tipo de instalación . . . 83
3.16. Listo para instalar . . . 84
3.17. Instalación completa . . . 84
3.18. Configuración servidor MySQL . . . 85
3.19. Inicio de configuración . . . 86
3.20. Acuerdos de licencia . . . 86
3.21. Selección de producto y características . . . 87
3.22. Ubicación de la instalación . . . 88
3.23. Proceso de instalación . . . 89
3.24. Configuración del servidor . . . 90
3.25. Tipo de conexión del servidor . . . 91
3.26. Contraseña del servidor . . . 91
3.27. Servicio de Windows . . . 92
3.28. Configuración completa . . . 93
3.29. Sentencia SQL para tabla de Alumno . . . 95
3.30. Sentencia SQL para tabla UA . . . 95
3.31. Sentencia SQL para tabla Curso . . . 96
3.32. Sentencia SQL para tabla Calificacion. . . . 97
3.33. Sentencia SQL para tabla Historial . . . 97
3.34. Icono de conexión . . . 98
3.35. Cargardo el editor de SQL . . . 99
3.36. Editor SQL . . . 100
3.37. Componentes Editor SQL . . . 101
3.38. Ejemplo de consulta búsqueda de alumnos por número de cuenta . . . 102
3.39. Ejemplo del resultado de la búsqueda de alumno por número de cuenta . . . 102
3.40. Ejemplo de consulta buscar alumnos por Apellido Paterno . . . 103
3.41. Ejemplo del resultado de la Consulta . . . 103
3.42. Ejemplo de consulta de búsqueda de alumno por Apellido Materno . . . 104
3.43. Ejemplo del resultado de la Consulta . . . 104
3.44. Ejemplo de consulta búsqueda de alumno por Nombre . . . 105
3.45. Ejemplo del resultado . . . 105
3.46. Consulta de búsqueda de alumno por periodo . . . 106
3.47. Resultado de la consulta . . . 106
3.48. Ejemplo de consulta buscar alumno por UA . . . 107
3.49. Ejemplo de resultado de consulta . . . 107
3.50. Ejemplo de consulta de alumnos por grupo . . . 108
3.51. Ejemplo de resultado de la búsqueda de alumnos por grupo . . . 108

3.52. Ejemplo de consulta de alumnos por Turno . . . 109
3.53. Ejemplo de resultado de la consulta Alumnos por Turno . . . 109
3.54. Ejemplo de la consulta alumnos por año de ingreso . . . 110
3.55. Ejemplo de resultado de la consulta Alumnos por año de ingreso . . . 110
3.56. Ejemplo de la consulta promedio de alumno . . . 111
3.57. Ejemplo de resultado de promedio de alumno . . . 111
3.58. Ejemplo de consulta promedio por grupo . . . 112
3.59. Ejemplo de resultado de promedio por periodo . . . 112
3.60. Ejemplo de consulta promedio por UA . . . 113
3.61. Ejemplo de resultado de promedio por UA . . . 113
3.62. Ejemplo de consulta estado académico de alumnos . . . 114
3.63. Ejemplo de resultado de estado académicos . . . 114
3.64. Ejemplo de consulta alumnos inscritos por periodo . . . 115
3.65. Ejemplo de resultado de Alumnos inscritos por periodo . . . 115
3.66. Ejemplo de consulta alumnos inscritos por UA . . . 116
3.67. Ejemplo de resultado alumnos inscritos por UA . . . 116
3.68. Ejemplo de alumnos inscritos en linea de acentuación . . . 117
3.69. Ejemplo de resultado de Alumnos inscritos por línea de acentuación . . . 117
3.70. Ejemplo de consulta de calificaciones por número de cuenta . . . 118
3.71. Ejemplo de consulta de resultado de calificaciones por número de cuenta . . . 118

4.1. Interfaz de sistema para control de alumnos . . . 121
4.2. Interfaz del sistema de control de alumnos. . . . 121
4.3. Consulta en la interfaz . . . 122
4.4. Resultado de la búsqueda. . . . 122
4.5. Ventana de información: Ingresar número de cuenta . . . 123
4.6. Venta de información: Elija una opción. . . . 124

Índice de cuadros

2.1. Símbolos E-R [27]. . . . 34
2.2. Operadores aritméticos y lógicos [14] . . . 48
2.3. Valores para los campos . . . 49

3.1. Modelo E/R y Modelo relacional . . . 60
3.2. Relación Alumno e Historial . . . 68
3.3. Relación Alumno y Curso. . . . 69
3.4. Relación UA y Curso. . . . 70
3.5. Relación UA e Historial. . . . 71
3.6. Relación Curso y Calificación . . . 72
3.7. Descripción atributo NumeroCuenta . . . 73
3.8. Descripción atributo ApellidoP . . . 73
3.9. Descripción atributo ApellidoM . . . 73
3.10. Descripción atributo Nombres . . . 74
3.11. Descripción atributo CURP . . . 74
3.12. Descripción atributo TelParticular . . . 74
3.13. Descripción de atributo TelCelular . . . 74
3.14. Descripción atributo Email . . . 74
3.15. Descripción atributo NoSeguroSocial . . . 75
3.16. Descripción atributo AnoIngreso . . . 75
3.17. Descripción atributo Periodo . . . 75
3.18. Descripción atributo Estado acedemico . . . 75
3.19. Descripción atributo ClaveUA . . . 76
3.20. Descripción atributo NombreUA . . . 76
3.21. Descripción atributo HorasTeoricas . . . 76
3.22. Descripción atributo HorasPracticas . . . 76
3.23. Descripción atributo CreditosTotales . . . 76
3.24. Descripción atributo . . . 77
3.25. Descripción atributo Subsecuente . . . 77
3.26. Descripción atributo Nucleo . . . 77
3.27. Descripción atributo IdUAOfertada . . . 77
3.28. Descripción atributo Periodo . . . 78

3.29. Descripción atributo Grupo 78
3.30. Descripción atributo Turno 78
3.31. Descripción atributo IdHistorial 78
3.32. Descripción atributo PeriodoCurso 79
3.33. Descripción atributo CalificacionFinal 79
3.34. Descripción atributo IdLinea 79
3.35. Descripción atributo Baja 79
3.36. Descripción atributo BajaTotal 79
3.37. Descripción atributo IdCalificación 80
3.38. Descripción atributo PrimerParcial 80
3.39. Descripción SegundoParcial 80
3.40. Descripción atributo Final 80
3.41. Descripción atributo ETS 81
3.42. Descripción de atributo EEXTS 81

Capítulo 1

Introducción

Las licenciaturas ofertadas en el Centro Universitario UAEM Zumpango (CUZ), al igual que las instituciones educativas en México se ven obligadas a llevar un seguimiento del estado académico de sus estudiantes inscritos, por lo que generalmente los responsables de cada programa educativo generalmente llevan su propio seguimiento, a menudo usando hojas de cálculo, por que es la forma más conocida de llevar este control.

La Universidad Autónoma del Estado de México (UAEM), del CUZ es una institución con un modelo educativo basado en competencias. Esto significa que se forma a los alumnos no sólo con conocimientos, sino también se les desarrollan habilidades, actitudes y se les inculcan valores. El seguimiento académico de alumnos de programación educativos en instituciones como la UAEM, es una tarea complicada. Esto en debido a que se deben de considerar varios aspectos involucrados. Por ejemplo, en el CUZ, la matricula de alumnos del Programa Educativo de Ingeniería en computación (PE de ICO) está compuesta de dos turnos, matutino y vespertino. El PE de ICO en el CUZ cuenta con tres laboratorios equipados para el desarrollo de las competencias de los alumnos, estos laboratorios son: Laboratorio de Redes de Comunicación y Sistemas Distribuidos, Laboratorio de Electrónica y Arquitectura de Computadoras, Laboratorio de Programación e Ingeniería de Software.

El plan de estudios de PE de ICO se compone de 58 Unidades de Aprendizaje (UA) obligatorias y de 6 a 9 UA optativas. En total, un egresado debe de haber cursado y aprobado de 64 a 67 UA, para completar un total de 430 a 450 créditos, dependiendo de la línea de acentuación elegida. Existen cuatro líneas de acentuación para que el alumno elija una, esta son: Administración de proyectos informáticos, Redes y comunicaciones, Interacción hombre-máquina e inteligencia computacional y Desarrollo de software de aplicación. El mapa curricular cuenta con ocho áreas curriculares que son las siguientes: Entorno social, Matemáticas, Arquitectura de computadoras, Redes, Software de base, Tratamiento de la información, Programación e ingeniería de software e Interacción hombre - máquina.

Actualmente, existen tres planes vigentes en el PE de ICO del CUZ, estos son F2, F3 y F4. Los alumnos de recién ingreso tienen como plan el F4, mientras los próximos a egresar tienen el plan F2. Llevar el seguimiento académico de todos los estudiantes es laborioso. Aunque el sistema informático de de control escolar lleva la administración de calificaciones, no es capaz de generar todos los indicadores requeridos para las evaluaciones externas que se le realizan a los programas educativos.

El problema en el que este trabajo se centra es en el diseño de una base de datos para el seguimiento académico de alumnos del PE de ICO del CUZ. Las consultas implementadas pueden generar los siguientes reportes: índices de reprobación, listas de alumnos inscritos (por periodo, por UA), lista de titulados, estadísticas sobre UA con más altos indices de reprobación y aprobación, situación académica actual de cada alumno.

El resto del documento se divide en 5 capítulos. El en Capítulo 2 se describen las tecnologías utilizadas para el desarrollo del sistema, en el que se presentan las bases teóricas que respaldan esta tesis; el Capítulo 3 se encuentra el desarrollo del sistema, en el cual se describe los pasos del diseño, se desarrolla una de varias soluciones que pueden surgir para resolver este el problema planteado. También de describe la forma y cada paso para el desarrollo de la BD; el Capítulo 4 se trata de las pruebas que se realizaron a la BD. Las conclusiones y trabajos futuros se encuentran en el Capítulo 5.

1.1. Planteamiento del problema

Aunque el sistema informático se basa sobre el seguimiento académico de los alumnos en un programa educativo, es una actividad laboriosa que consume una cantidad considerable de tiempo y esfuerzo. El contar con un sistema que automatice la generación de este tipo de reportes, permite tener una organización y control más específico y que este proceso sea más eficiente, además dese prevenir posibles errores de control escolar de la UAEM, y es capaz de atender las necesidades generales la Universidad, no fue diseñado para atender las evaluaciones que se le realizan a los programas educativos por organismos externos, como CIEES o CACEI. Estas evaluaciones solicitan presentar información específica sobre algunos indicadores por cohorte, tales como el de aprobación, retención, egreso y titulación, entre otros. Al no contar actualmente con un sistema que automatice el cálculo de estos indicadores, se tienen que realizar de manera manual, lo que representa una inversión de tiempo considerable, además de introducir cierto riesgo de error humano.

El problema que se resuelve en esta tesis, es la automatización del cálculo de algunos indicadores académicos para PE de ICO del CU UAEM Zumpango. Para esto, se propone el diseño completo de una base de datos, y la creación de las consultas SQL necesarias.

1.2. Motivación

Las tendencias actuales en materia de educación en México obligan a las instituciones educativas a tener actualizadas los indicadores académicos de sus programas educativos. Con el fin de mejorar la calidad educativa, las instituciones deben no sólo de mantener estos datos actualizados, sino también deben usarlos para identificar las fortalezas y debilidades académicas.

El plan de estudios del PE de ICO es flexible, lo que permite a los alumnos cursar UA en uno o diferentes espacios académicos, adelantar UA o darlas de baja en caso de ser necesario. Llevar un seguimiento académico en estos programas no es una tarea simple. Aunque es posible llevar un control manual o usar hojas de cálculo, esta forma de procesamiento ya no es suficiente; la generación de reportes que puedan ser utilizados por coordinadores, requiere de un procesamiento eficiente, en el que se involucran búsquedas y consultas complejas.

1.3. Hipótesis

La generación de reportes sobre el seguimiento académico de los alumnos en un programa educativo es una actividad laboriosa que consume una cantidad considerable de tiempo y esfuerzo. El contar con un sistema que automatice la generación de este tipo de reportes, permite que este proceso sea más eficiente, además de reducir posibles errores humanos durante los cálculos.

1.4. Justificación

El seguimiento académico de los alumnos inscritos en el PE de ICO del CUZ resulta una tarea bastante laboriosa. Esto debido a que los archivos generados por el sistema de control escolar de la UAEM son de texto plano, en un formato que no es tan fácil de importar a hojas de cálculo o a una base de datos. Lo anterior, hace necesaria la aplicación de un proceso manual para la captura de datos, para posteriormente realizar los cálculos de los indicadores.

Por otra parte, se investigó sobre la existencia de sistemas que pudieran cubrir estas necesidades, y se encontró que aunque existen sistemas de control de alumnos, estos se encuentran orientados principalmente a escuelas de educación básica y media superior, y no permiten generar reportes como los solicitados en las evaluaciones externas.

1.5. Objetivos generales y específicos

1.5.1. General

Diseñar e implementar las tablas, relaciones y consultas de una base de datos sobre la información académica de los alumnos del PE de ICO del CUZ, para automatizar el cálculo de los principales indicadores académicos del programa.

1.5.2. Específicos

- Analizar el plan de estudios y el mapa curricular del PE de ICO CUZ, para identificar seriaciones entre UA, áreas, créditos, y otros elementos.
- Obtener los requerimientos respecto al tipo de reportes necesarios para llevar a cabo el seguimiento académico de los alumnos inscritos en el PE de ICO del CUZ.
- Diseñar las tablas y las relaciones que componen la base de datos.
- Diseñar e implementar las consultas SQL para responder a los requerimientos de reportes.
- Instalar las herramientas necesarias y configurarlas.
- Implementar la base de datos y alimentarla con algunos datos de prueba.
- Realizar pruebas de funcionamiento de la base de datos y de los reportes generados.

1.6. Alcance

La base de datos es creada en MySQL Workbench sobre el sistema operativo Windows. Los requerimientos de las consultas fueron consultadas con el Coordinador actual del PE de ICO. Las consultas se implementaron en SQL dentro del mismo Workbench, y se ejecutan de manera local. La interfaz gráfica propuesta es para consultas fue generada en Java con la biblioteca Swing.

Capítulo 2

Tecnologías utilizadas para el desarrollo del sistema

Las bases de datos se encuentran presentes en la mayoría de las aplicaciones actuales. Los negocios de todo tipo utilizan sistemas de información que contienen como elemento principal a una base de datos, en la que almacenan información relacionada con sus clientes, proveedores, y procesos administrativos, entre muchos otros. Este capítulo está dedicado a la explicación de los fundamentos de las bases de datos, comenzando desde su historia hasta los conceptos de bases de datos relacionales.

2.1. Historia de las bases de datos

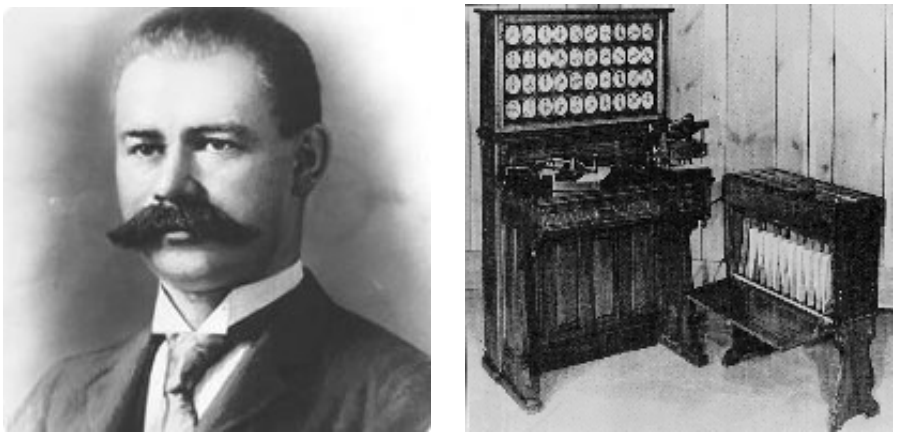
Desde la antigüedad, el uso de las bases de datos se ha hecho presente, ejemplo de ello son los negocios mercantiles, donde en el oriente se crearon fichas de arcilla usadas para llevar el registro de la mercancía, las cosechas, el número de piezas de pan, la cantidad de ovejas y prendas de vestir. Otro método para llevar un control de sus cuentas, era haciendo cortes y muescas en palos de madera, o inclusive, se realizaban nudos en cuerdas. Estos métodos, aunque eran muy primitivos, demuestran el inicio de los registros que contenían las ventas o compras.

En el siglo XVII, Blaise Pascal creó una máquina mecánica basada en engranes entrelazados, que podía procesar datos, con el fin de ayudar a su padre en la recolección de impuestos. En 1805, Joseph Marie Jacquard desarrolló el primer telar programable con tarjetas perforadas. Aunque su telar no era un dispositivo para realizar cálculos, sí era capaz de almacenar datos, por lo que se considera un aporte importante en la historia de las bases de datos primitivas.

Charles Babbage, quien retomó el trabajo de Jacquard, creó una máquina diseñada para el almacenamiento mediante las tarjetas perforadas, aunque dicha máquina no la terminó completamente, sí estaba diseñada para almacenar instrucciones de cálculo, realizar operaciones y producir un resultado.

A finales del siglo XIX, Herman Hollerith (véase la Figura 2.1) basado en el trabajo de tarjetas perforadas de Jacquard, construyó dispositivos que permitieron que en el censo de 1890, el conteo total del censo de población en Estados Unidos se terminará un mes después de recopilarlos, estos

dispositivos fueron clasificados como electromecánicos, porque utilizaban la electricidad y contadores mecánicos que en conjunto con la electricidad hacían la tabulación para la concatenación de los datos. En esta época se dio inicio a la era del procesamiento de información, esto debido a los pequeños avances tecnológicos de maquinaria que en esa época se presentaba [7]. Poco después, Hollerith fundó la Tabulating Machine Company, esta compañía en conjunto con otras, es actualmente la International Business Machines Corporation (IBM).



Fuente: [7]

Figura 2.1: Herman Hollerith y máquina tabuladora

En la década de los cincuenta surgen las cintas magnéticas, que sirvieron para automatizar las nóminas, almacenadas en cintas, que se convirtieron el medio dominante de almacenamiento. La desventaja era que sólo se podían leer secuencial mente. A mediados de está época se introdujeron las computadoras electrónicas.

En los años sesentas aparece la expresión "bases de datos", y con ello los discos magnéticos, estos fueron el reemplazo de las cintas magnéticas y fueron considerados un medio masivo de almacenamiento de información y elimina por completo la desventaja de las cintas magnéticas, ya que la información puede ser consultada en cualquier punto del disco. A mediados de los sesentas comenzó la conversión de cintas a discos magnéticos. IBM desarrolló el Sistema de Gestión de la Información (IMS, por sus siglas en inglés *Information Management System*) [22], construido como estructura alternativa para la representación de los datos denominadas bases de datos jerárquicas y en red. También, en esta década, apenas diez años después de la introducción de las computadoras electrónicas, la necesidad de privacidad, respaldo y recuperación de la información llevó a requerir un software para la administración de los datos y hardware mucho más rápido. Esto dio origen al Sistema Gestor de Base de Datos (DBMS por sus siglas en inglés *Data Base Managment System*. En español, se conoce como SGBD.).

En 1970, el científico informático Inglés Edgar Frank Codd publicó una serie de reglas para la administración de los sistemas de base de datos relacional en un documento titulado: "*A relational*

model of data for Large Shared Banks"[24]. En él se define el modelo relacional, aún que no tuvo gran impacto ya que carecía de características que el modelo de red y jerárquico en ese tiempo dominaban. IBM desarrolló un proyecto llamado *System R,* que fue el primer sistema que implemento el modelo de Codd, considerado también como la base de SQL/DS que desarrollo técnicas para los sistemas de Bases de Datos (BD) relacional. El fundador de la BD en Oracle, Larry Ellison, desarrolla un sistema de administración de base de datos, esto se logró a partir de los avances que proporcionó Codd y su modelo relacional.

En la década de los 80, las bases de datos relacionales compitieron con las BD jerárquicas y de red en cuestión de rendimiento, las BD relaciones se convirtieron un elemento dominante por su fácil uso y poco a poco fueron desplazando a las BD jerárquicas y en red. En está década también se dio a conocer la investigación y trabajo de las BD orientadas a objetos, así cómo las BD paralelas y distribuidas [21].

En la década de los 90, las BD orientadas a objetos (BDOO) ocuparon un lugar importante, ya que estas han tenido gran éxito al gestionar datos complejos, detalle que las BD relacionales no han podido solucionar.

2.2. Bases de datos

La información anteriormente se organizaba en "archivos planos", este tipo de archivos sólo contienen filas y columnas (campo y registro respectivamente) sin ningún formato ni relaciones entre campos ni registros. En la siguiente subsección se abordarán con mayor profundidad las bases de datos actuales. Hoy en día, el avance tecnológico permite la gestión automática de datos mediante sistemas de información (SI). Algunos ejemplos en donde se utilizan este tipo de sistemas son negocios, administración de los gobiernos y educación. Los SI utilizan bases de datos que ayudan al procesamiento, organización y almacenamiento de grandes volúmenes de datos.

El término de base de datos surge en los años sesentas en Estados Unidos (EEUU), los desarrolladores de software denominaron Bases de Datos (BD) a sus sistemas que se encargaban de almacenar los datos, sin importar que debían cumplir con algunas propiedades. Estas propiedades fueron creándose e incorporándose a los sistemas de acuerdo al desarrolló que se fue generando en los sistemas de administración de la información [26]. Una definición actual de las BD es la siguiente:

"*Una base de datos es una colección de archivos relacionados que almacenan tanto una representación abstracta del dominio de un problema real cuyo manejo resulta de interés para una organización, como los datos correspondientes de la información acerca del mismo.* [26]"

Una BD es considerada como un armario de datos que se encuentran relacionados entre sí, esta colección de datos se almacena en archivos electrónicos, de manera que permite a usuarios tener un acceso rápido a los datos, además de proporcionar mecanismos para mantenerlos organizados y consistentes. Un objetivo de las BD datos es eliminar la redundancia de los datos.

Los sistemas de base de datos (SBD) son sistemas computarizados que se encargan del almacenamiento de información, permitiendo que se puedan recuperar, actualizar por medio de paquetes de software llamados sistemas manejadores de base de datos (DBMS). La información que se maneja es

importante para cada usuario que la genera. Los SBD comprenden cuatro componentes principales: datos, hardware, software y usuarios.
Los datos en los SBD se encuentran en equipos personales o en grandes equipos como son los Mainframes, los sistemas grandes tienden a ser **multiusuarios**, los equipos pequeños son de **monousuario**. Los sistemas multiusuarios son aquellos en los que varios usuarios tienen acceso de manera simultánea y los sistemas de un solo usuario se accede en momentos determinados. En los sistemas grandes, los datos generalmente son integrados y compartidos. El término de integrado se entiende que es la unión de muchos archivos, que es la eliminación de redundancia de datos, para esto se utilizan las referencias. El término de compartido quiere decir que se puede tener acceso simultáneo a datos individuales, esto también se le denomina "acceso concurrente" o bien acceso a datos en el mismo tiempo [4].

El hardware cuenta con los siguientes componentes: dispositivos de almacenamiento de acceso directo, computadora central y complementos asociados como la memoria principal, procesador de entrada y salida (E/S) y la unidad de control. Los aditamientos se refiere a los dispositivos de almacenamiento secundarios (almacenamiento externo como cintas magnéticas), el acceso a este tipo de dispositivos es más lento por el movimiento electromecánico (mili segundos), el costo de estos dispositivos es mucho menor en comparación de la memoria principal, además de utilizarlo para almacenar datos permanentes independientemente del problema que representa el acceso lento a estos dispositivos. La memoria principal, que se utiliza para el almacenamiento temporal de los datos al ser ejecutados por el CPU, proporcionan el acceso a la memoria principal en menos de un micro segundo [29].

Otro de los componentes de los SBD es el Software, en particular para el almacenamiento de datos, se usan sistemas de administración de base de datos (DBMS). El DBMS controla el acceso a la BD, por lo tanto, el objetivo principal es ocultar a los usuarios de la base de datos los detalles al nivel de hardware.

Existen tres tipos de usuarios: los programadores de aplicaciones, los usuarios finales y el administrador de base de datos (DBA). El primero es el responsable de la programación de aplicaciones para la BD bajo cualquier plataforma o lenguaje de programación, estas aplicaciones van de la mano con el SGBD, esto se da a través de instrucciones de SQL. Los usuarios finales son los que interactúan con el sistema. El administrador de base de datos DBA es un profesional que consiste en crear la BD también es responsable que el sistema funcione correctamente.

Las BD han evolucionado a través del tiempo, por lo que hay varios tipos y modelos de ellas. A continuación se presentan los modelos y tipos más importantes.

2.2.1. Tipos de bases de datos

De manera similar a los tipos de computadoras, no existe un tipo modelo, en las que no exista u único modelo o marca, con las base de datos, ocurre los mismo: existen varios modelos y cada modelo puede tener usos diferentes, de acuerdo a las necesidades de los usuarios. La manera en que esta divididas las BD sera de acuerdo a su función, desde las mas antiguas hasta las usadas o más avanzadas.

2.2.2. Modelos de bases de datos

"El modelo de base de datos es considerado como un conjunto de ideas lógicas utilizadas para representar la estructura de los datos y las relaciones entre ellos" [20]. Estos modelos agrupan en dos categorías: conceptuales y de ejecución .

El modelo conceptual facilita la representación del conjunto de datos, por lo que sus conceptos son cercanos al mundo real, este modelo incluye el modelo de Entidad-Relación (E-R) y el modelo Orientado a Objetos. El modelo conceptual utiliza relaciones para describir la asociación que existe entre los datos: uno a muchos, muchos muchos y uno a uno. Las notaciones taquigraficas serían las siguientes: 1:M, M:N, 1:1, respectivamente.

El modelo de ejecución se enfoca en cómo están representados los datos en la base de datos. Este modelo incluye el modelo de base de datos jerárquico, de red, el relacional y el orientado a objetos [20].

2.2.2.1. Base de datos jerárquica

El modelo jerárquico a principios de los años 60's, los datos en él están organizados como un árbol invertido. En el diagrama de árbol, la relación es de uno a muchos (1:M), en otras palabras, hay un único camino de acceso a los datos [9]. Este modelo básicamente establece niveles entre los campos de registro, donde los nodos corresponde a los campos y las ramas a un registro. Para localizar un campo en algún nivel, es necesario ubicarse en el nivel superior y decender por las ramas, hasta llegar al campo requerido [14].En la Figura 2.2 se muestra un ejemplo de las BD jerárquicas.

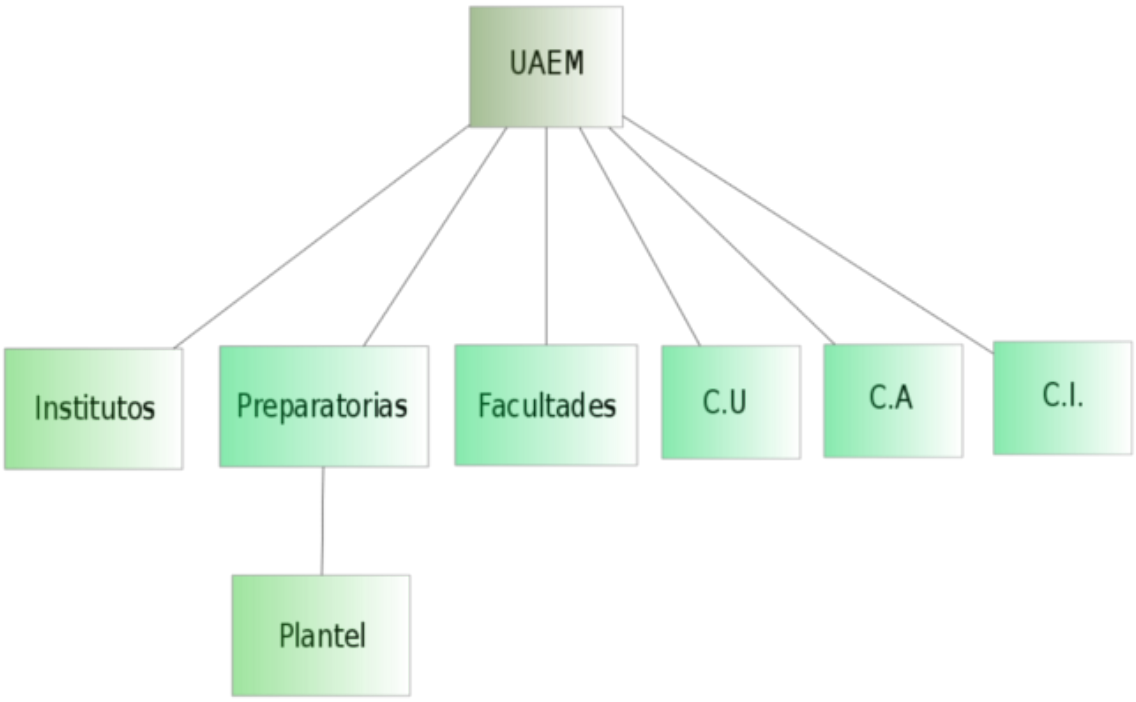


Figura 2.2: Modelo Jerárquico

El árbol es considerado como una sola entidad. Los elementos que lo conforman son dos, el miembro superior, llamado también como raíz o padre, los miembros de nivel inferior, son denominados hijos. Los elementos que ocupan el mismo nivel se les llama gemelos.

Ventajas

Las ventajas de las BD jerárquicas fueron importantes en los años 60's, ya que a partir de ellas se tomó como fundamento las BD actuales, ahora se mencionan algunas de sus ventajas:

- Simplicidad conceptual: simplifica el diseño, esto es debido a la relación, es decir, es más sencillo visualizar las BD conceptualmente para simplificar el diseño.
- Seguridad en el diseño de la base de datos: el Sistema manejador de la base de datos realiza la ejecución uniformemente sin tener que recurrir al programador.
- Independencia de datos: disminuye el mantenimiento del programa debido a que el SGBD puede crear la independencia de los datos.
- Integridad de las bases de datos: el modelo jerárquico promueve la integridad de los datos, dado que mantiene los vínculos.
- Eficiencia. es eficiente cuando la BD contiene grandes volúmenes de relaciones 1:M y cuando se requieren demasiadas transacciones [20].

Desventajas

- Ejecución compleja: debido al almacenamiento físico de los datos y los problemas de dependencia de los datos.
- Difícil de administrar: Cualquier cambio o re ubicación de segmentos, será necesario realizar un cambio en todos los programas de aplicación de la base de datos, es por eso que la administración de base datos será complicada.
- Carece de independencia estructural: La base de datos jerárquica se caracteriza por ser un sistema navegación, el programador debe conocer las rutas de acceso pertinentes para ingresar a la base de datos.
- Complejidad de la programación y uso de las aplicaciones: los programadores y usuarios finales deben de conocer como están distribuidos físicamente los datos para poder ingresar a ellos, es por eso que se dice las bases de datos jerárquicas están echas por los programadores y para los programadores.
- Limitaciones de ejecución: las relaciones comunes no se ejecutan con el estándar 1:M.
- Falta de estándares: este modelo esta incorporado a todo el software de BD, la ejecución unicamente se ajusta un modelo y carecía de un lenguaje de definición de datos estándar, así como también no contaba con un lenguaje de manipulación, para manipular la BD [20].

Debido ha estas desventajas, las BD jerárquicas casi han desaparecido en la actualidad.

2.2.2.2. Base de datos en red

Este modelo se creó para representar relaciones más complejas, superando así a las BD jerárquicas. Además, se incluyeron las relaciones de (1:M) y (N:M). Las entidades son grafos, donde estos permiten el acceso a las otras entidades por varios caminos, sin tener que acudir a la raíz, por lo que la búsqueda en más rápida. Este modelo de BD también aporta un gran beneficio, ya que elimina el problema de la redundancia de datos, además este modelo es más utilizado por programadores aunque eso signifique una dificultad a la hora de administrar los datos en la base de datos [9]. Un ejemplo de bases de datos en red se puede ver en la Figura 2.3

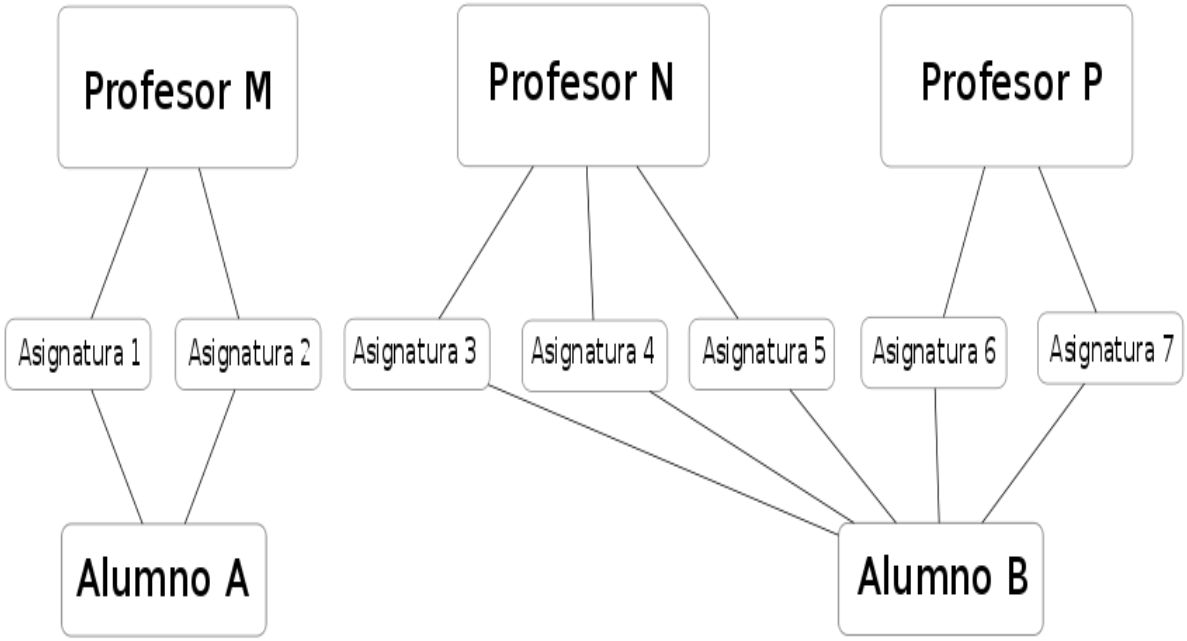


Figura 2.3: Modelo de base de datos en red [9]

Para tratar de establecer estándares la CODASYL (Conference on Data Systems Languages por sus siglas en Inglés) convoca a una reunión para establecer estándares, mismas que fueron aprobadas por ANSI y estas tuvieron gran éxito que se creó el Database Task Group (DBTG) que se encargaba de definir estándares para facilitar la creación y manipulación de las BD.

El modelo de BD en red, se puede decir que es muy parecido al modelo de BD jerárquico, aunque se conservan muchas de las ventajas de este modelo, el tiempo se encarga de corregir estos aspectos.

Ventajas

- Simplicidad conceptual. Es sencilla y simple el diseño de la vista conceptual como en el modelo jerárquico.
- Maneja más tipos de relaciones. En el modelo de red es más facil el manejo de relaciones como

M:N que en las BD jerárquicas.

- Flexibilidad de acceso a los datos. Es mas fuerte la flexibilidad de acceso a los datos, se puede tener acceso a registros dentro de un conjunto de propietarios.
- Promueve la integridad de las BD. Define registros propietario y después el miembro, el miembro no puede existe, si no tiene un propietario.
- Independencia de datos. Tiene una idependencia suficiente para aislar los programas de los detalles de almacenamiento.
- Cumplimiento de estándares. Basadas en los estándares de los años 70's que incluyen los DDL y DML que facilitan la administración y portabilidad de datos [20].

Desventajas

- Complejidad del sistema. El acceso es navegacional por los tanto se puede acceder a los registros una sola vez, esto significa un gran problema para los programadores y usuarios finales, ya que deben conocer a la perfección la BD.
- Falta de independencia estructural. Depende de una estructura por el tipo de acceso de navegación, falla la independencia estructural si se cambia la estructura, por otra parte los programas tendrian que ser revalidados antes de accesar a la BD [20].

2.2.2.3. Base de datos orientada a objetos

Las BD orientadas a objetos (BDOO) en concepto surgieron en los años 70, desarrollada en el centro de investigación de Xerox, permitiendo que en los años 90 se impulsara mucho gracias al avance de la tecnología y más en el campo de la computación, el suministro de generar y mantener una base de datos en una computadora con programas más complejos[20].

Las BDOO surgieron como una respuesta a las necesidades de manejar datos con una estructura compleja, y con operaciones específicas.

A diferencia de otros tipos de BD, en los que sólo es posible diseñar la estructura y las relaciones entre los datos, en las BDOO se pueden también definir las operaciones que se les pueden aplicar [23]. Otra ventaja es la posibilidad de definir nuevos tipos de datos. Por ejemplo, es posible crear un tipo de dato Persona en un BDOO, y asignar una relación con una entidad Empleado (es un tipo de Persona). A la entidad Empleado, podría generarle una operación establecerSalario(), que sólo acepte valores en un cierto rango.

Una desventaja de las BDOO respecto a las BDR, es que no pueden realizarse consultas en las que no se haya establecido previamente una relación entre entidades, cosa que sí es posible en las concatenaciones (joins) entre tablas sin relaciones en las BDR [12].

Es importante mencionar que las BDOO no son un reemplazo de las BDR, es decir, las primeras no pueden usarse en todas las aplicaciones en las que se usan las BDR. Algunas aplicaciones exitosas de las BDOO son simuladores, sistemas de diseño asistido por computadora y sistemas de información geográfica.

La estructura básica de una BDOO se basa en lo siguiente:

- Los objetos son entidades o eventos del mundo real.
- Los atributos se describen como las propiedades que tiene un objeto.
- Los objetos contienen ciertas características que se agrupan en clases.
- Las clases se caracterizan por se organizadas en jerarquías [20].

Ventajas

- Agrega contenido semántico: es contenido con mayor significado, donde se relacionan las entidades.
- Presentación visual: las BDOO es más fácil visualizar las relaciones más complejas, por su contenido semántico.
- Integridad: incluye herencia de manera que protege los datos y además tiene relaciones complejas.
- Independencia estructural: garantiza la integridad y la independencia estructural de los datos [20].

Desventajas

- Falta de estándares: porque no existe un método para acceder a los datos.
- Acceso navegacional: este es el método que poseen las BDOO parecido a las de red y jerárquico.
- Curva de aprendizaje pronunciada: esto es por la ausencia de estándares y por el método de navegación de los datos, hace que los datos sean complejos y se hace difícil diseñar y ejecutar para los usuarios finales.
- Complejidad del sistema: la complejidad del sistema hace que las tracciones sea lenta, y cuando son lentas es desagradable trabajar los datos[20].

2.2.2.4. Base de datos operacionales o transaccionales

Los almacenes de datos se han convertido en almacenes de datos activos, esto por que vinculan el almacén con los sistemas operacionales de los usuarios. Existen niveles de operación si se comunican proveedores con clientes para fines de información del negocio.

Ahora la información se almacena en BD, en el del trabajo diario como lo es en la industria, transaccionales *Son bases de datos cuyo único fin es el envío y recepción de datos a grandes velocidades, por lo que las redundancia y duplicación de información no constituye un problema, como ocurre en las demás bases de datos* [10].

Este tipo de base datos se utiliza para el trabajo transaccional diario de los sistemas de información de las empresas, conocido como OLTP (On-Line Transactional Processing), realizando el análisis de trabajo en tiempo real sobre la OLAP (On-Line Analytical Processing), pero este proceso esta diseñado para el análisis no para el proceso, entonces esto daña el proceso transaccional, este trabajo unicamente se realiza en tiempo donde no se use la BD [15].

2.2.2.5. Bases de datos relacionales

El modelo relacional, desarrollado por Edgar Frank Codd en el año de 1970, que analizó el modelo, produjo una BD conceptual que permitió cambios importantes en la evolución de las BD. La estructura básica de este modelo es que realiza funciones más fáciles de entender y ejecutar[20].

Las bases de datos relacionales, se basan en el modelo relacional, usando tablas en términos informales son las relaciones de cada una de las tablas. En cada tabla y nombre de las columnas de la base de datos es importante por permiten interpretar los valores que se encuentran en cada tabla, es decir, "los valores se pueden interpretar como hechos que describen una entidad". El modelo relacional determina a una tabla como una *relación*, una fila como *tupla*, una cadena de caracteres como un *atributo* y los tipos de datos como *dominios*.

Un *dominio* son los valores indivisibles en el modelo relacional, el nombre del dominio ayuda a identificar los valores, algunos ejemplos de dominio son: Números telefónicos, números de seguro social, nombres, promedios de notas, edades de empleados, departamentos académicos, etc.

Este tipo ofrece varias ventajas importantes a diferencia de las BD en red y jerárquicas.

Ventajas

- Independencia estructural: no se utiliza métodos de acceso navegacional, no afectan los cambios la estructura de la BDR
- Se puede realizar consultas complejas, utilizando varias tablas, permitiendo emplear el lenguaea estructurado de consultas (SQL por sus siglas en Inglés).
- Actúan sobre las tablas y no en los registros como en otros sistemas [14].
- Simplicidad conceptual: el modelo relacional es aun más simple que el nivel conceptual.
- Diseño y ejecución: ofrece independencia de datos e independencia estructural permitiendo que la BD sea más fácil de diseñar y administrar.
- Optimo administrador de BD, tanto para diseñadores y usuarios [20].

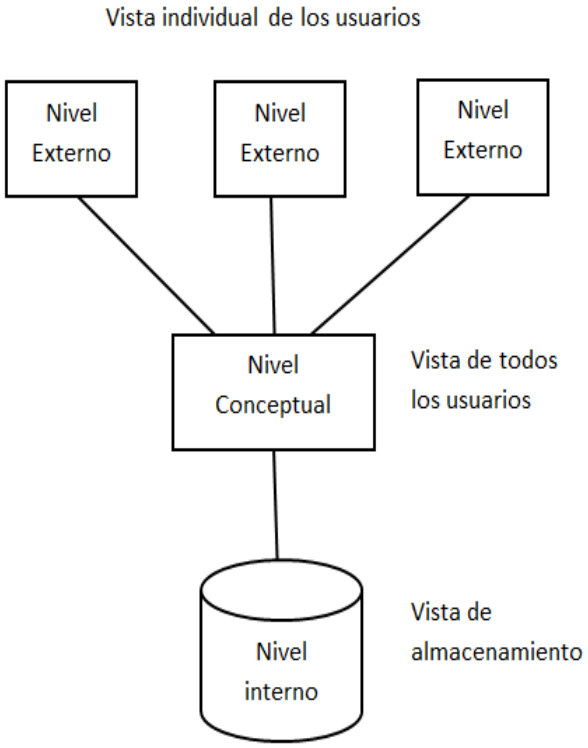
Desventajas

- Las BD relaciones tiene una gran desventaja, el costo de estas es muy elevado tanto la creación como la administración del sistema.
- Este tipo de BD puede promover que se conviertan en "islas de información", la cual consiste en compartir y utilizar la información fácilmente, pero el incremento de las BD que se crean por los usuarios puede generar incosistencia en los datos lo que lleva a tener problemas con la información.
- La pérdida de datos es un problema que presentan las BD relacionales, ya que al diseñar la BD algunas de las BD relacionales tienen límite en la longitud de sus datos [14].

2.3. Arquitectura de los sistemas de bases de datos relacionales

La arquitectura de los sistemas de BD es sin duda una infraestructura útil para describir los conceptos generales de la BD, estas arquitecturas suelen ajustarse en la mayoría de los sistemas y es más parecida a la propuesta por el comité ANSI/SPARC (American National Standard Institute-Standard Planning and Requeriments Committee) en 1975, quien propuso esta arquitectura de tres niveles, (ver Figura 2.4) y se describen en tres esquemas. El propósito de esta arquitectura es separar los niveles interno y externo de

la estructura física [24].



Fuente: [4]

Figura 2.4: Niveles ae Arquitectura de base de datos

- **Nivel interno o físico:** es el que se encarga de ver cómo están almacenados los datos físicamente , además de detallar el espacio de direcciones. La vista interna se describe por medio del modelo conceptual el cual describe los diversos tipos de registros, además de especificar lo índices que existen.

- **Nivel externo o lógico:** basado en el modelo interno, éste se aproxima a la representación de los usuarios, es decir como visualizan los datos los usuarios, tal como un programador o usuario final. Cada usuario tiene a su disposición un lenguaje para la realización de sus funciones. Para el programador, cualquier lenguaje de programación, por ejemplo: C++, JAVA, Phyton etc., para el usuario final generalmente es un lenguaje de consultas o bien programas con formularios o

menús. Este nivel se aproxima a una representación que es familiar para los usuarios, ya sean finales o desarrolladores. Para los usuarios finales, esto se traduce en que se pueden comunicar usando una interfaz gráfica. Para los desarrolladores, se permite el uso de lenguaje de consultas o incluso código en algún lenguaje de programación.

- **Nivel conceptual:** representa el contenido general de los datos, es decir, es una vista tal cual como son los datos. La vista conceptual está definida mediante un esquema conceptual, conteniendo definiciones de cada uno de los registros. Las definiciones del esquema conceptual contienen características de seguridad y de integridad [4].

2.3.1. Independencia de datos

La razón principal de la arquitectura es proporcionar la independencia de datos, es decir, que los niveles inferiores no son afectados por los cambios de los superiores. Existen dos tipos de independencias: lógicas y físicas. La lógica esta relaciona a la inmunidad de los modelos externos, estos cambios son como agregar nuevos tipos de registros, nuevas relaciones o nuevos elementos, esto debe de ser posible sin afectar la vista externa que existe. Un punto importante es que los datos no deben actualizar cuando se realizan cambios en el nivel lógico.

La independencia física se refiere a la inmunidad del modelo lógico a los cambios en el modelo interno, es decir, cambios del método de acceso, cambio de algoritmo, uso de dispositivos nuevo y el cambio de estructuras deben tener efecto en el nivel lógico [24].

2.4. Administración de BD

Un administrador de base de datos es el encargado del diseño, operación y gestión de una BD. El administrador debe tener la habilidad de un buen comunicador y experto en habilidades interpersonales. Para la gestión de la BD es necesario planificar, coordinar y realizar tareas de proyectos para la BD y el personal. También un administrador debe tener habilidades técnicas para enfrentarse a problemas de hardware y software. Las habilidades interpersonales son importantes para comunicarse con los usuarios, lograr negociaciones y mediador para posibles problemas con los usuarios. Las funciones del administrador dependerán del proceso del proyecto, pero las funciones principales son planificación o diseño, desarrollo y gestión, como se muestra en la Figura 2.5, y que en seguida se describen [24].

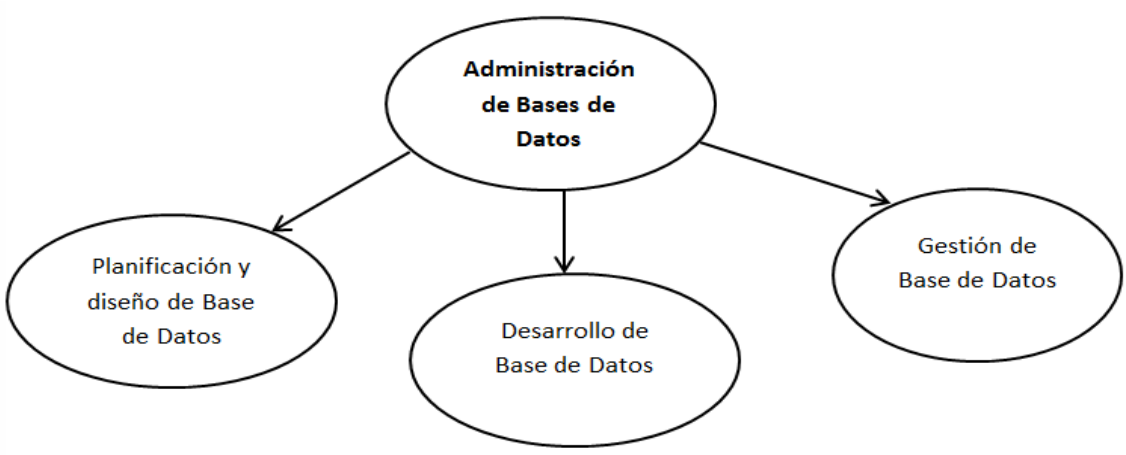


Figura 2.5: Administración de BD

2.4.1. Diseño de base de datos

En esta sección, se presentan de manera general el proceso de diseño de una BD, incluyendo además la terminología empleada en el área.

El diseño lógico es el punto de partida para poder generar una BD. El objetivo del diseño lógico es obtener una representación eficiente de los recursos para la estructuración de datos, así como para el modelado de restricciones disponibles en el modelo lógico.

En el diseño lógico, se toma en cuenta el esquema conceptual de la BD, el tamaño de la BD, la información que se tenga respecto a las consultas y transacciones que se realizarán, así como la frecuencia de las mismas. También se considera el rendimiento, que incluye el tiempo de respuesta medio, espacio de almacenamiento y uso de recursos tales como tiempo de procesador y tiempo de entrada salida.

El diseño de una BD se basa principalmente en observaciones de los requerimientos. El proceso general para diseñar una BD se divide en seis puntos que se resumen a continuación. Los tres primeros son importantes para el modelo relacional [22]:

- **Análisis de requisitos:** en este primer paso es importante entender qué datos se van a guardar en la BD, la operaciones son requisitos de rendimiento, es decir, es un proceso donde se pone a estudio y a prueba la BD que esperan los usuarios.

- **Diseño conceptual de la base de datos:** es la información obtenida en el análisis de requerimientos para desarrollar una descripción de alto nivel de los datos, al mismo tiempo se proponen al cliente elementos importantes a almacenar en la BD, y que no habían sido considerados. Este paso requiere del modelo ER, que es un modelo de alto nivel empleado en el diseño de datos, creando una idea sencilla de los procesos que van a representar los datos.

- **Diseño lógico de las bases de datos:** se elige un SGBD para implementar el diseño de la BD, transformando el diseño conceptual en un esquema del modelo de BD del SGBD.

- **Refinamiento de los esquemas:** es el análisis de conjuntos para identificar los problemas y corregirlos.
- **Diseño físico de las bases de datos:** Se considera el trabajo que realiza la BD y se reafirmará su diseño para un rendimiento eficaz.
- **Diseño de aplicaciones y de la seguridad:** En este punto hay que conocer las entidades relacionadas con el proceso de las aplicaciones, definir y conocer el papel que representará cada entidad en cada uno de los procesos que realizará en la aplicación.

2.4.2. Desarrollo de base de datos

- **Creación y desarrollo de la base datos.** Después de desarrollar el modelo físico, usando el lenguaje de definición de datos, se crea una estructura de BD para el gestor de la BD. Esta estructura creará bibliotecas, cargará los datos físicos y cargará los datos a la BD
- **Desarrollo de vistas de usuario.** Satisface las necesidades de los usuarios, se puede generar una vista de diseño pero esta puede ir cambiando de acuerdo a cómo el usuario se acostumbra al sistema.
- **Escritura y mantenimiento de documentación.** De acuerdo a cómo avance el proyecto, la documentación se escribe mediante un diccionario de datos. El Administrador de la BD debe asegurar que la información sea precisa de acuerdo a la BD.
- **Desarrollo y fortalecimiento de estándares de datos.** Hay que establecer estándares en beneficio de los usuarios. Para insertar y actualizar los datos es necesario que se cumpla con un estándar, por ejemplo, arrojar valores predeterminados, rangos aceptables etc. Los sistemas pueden verificar errores y restricciones de rango, pero también antes de actualizar puede comprobar las restricciones de valores entre claves y las relaciones entre los valores de un registro, el mismo archivo y en diferentes archivos.
- **Desarrollo y fortalecimiento de estándares de programa de aplicación**. Son instrucciones de seguridad y privacidad de los datos, sujetas a auditoría. Están sujetos a facilitar el desarrollo de la aplicación. Estos estándares son pertinentes para antiguas y nuevas aplicaciones.
- **Desarrollo de procedimientos operativos.** El ABD establece procedimientos para arranque diario, corrido de operaciones, errores, respaldos, procedimientos de seguridad, registro de fallas de hardware y software, medición de desempeño en caso de fallas, reinicio y recuperación después de la falla. Estos procedimientos los realizar operadores capacitados.
- **Capacitación de usuarios.** Son sesiones de capacitación que deben tomar los usuarios finales, programadores de aplicaciones y programadores de sistemas, esto para que aprendan a usarla más adecuadamente [24].

2.4.3. Gestión de Base de Datos

En este apartado es totalmente responsabilidad y trabajo del administrador de la BD, a continuación se describen algunas responsabilidades [24]:

- **Monitoreo del desempeño:** Analizar estadísticas del desempeño de la BD, atender las quejas y sugerencias, medir tiempo de consultas y tiempo de ejecución del SGBD.
- **Ajuste y reorganización:** Si el desempeño no es óptimo el administrador puede agregar cambios, reorganizando archivos e incluso cambiar el dispositivo de almacenamiento por una más eficaz. Si el problema no se resuelve con los cambios anteriores es posible que se cambie el modelo físico y recargar la BD.
- **Nuevas versiones de BD:** Cuando existen actualizaciones del SGBD, el administrador debe de evaluar y probar si es necesario cambiar al nuevo producto [24].

2.5. Modelo relacional

Los modelos de entidad relación (E-R) están basandos en la teoría de conjuntos matemáticos, los cuales tienen diversos propósitos, entre ellos:

- Evitar la redundancia de los datos, para ello, estos datos son divididos en distintos grupos en los cuales no se duplican
- Satisfacer la consistencia de datos. Es decir, si existe un cambio en alguna de las tablas, entonces ese cambio se efectúa en toda la BD donde se vea afectado ese campo.

El modelo E-R se encarga de las observaciones reales lo cual indica que son objetos básicos, estos son denominados *entidades* y la relación entre los objetos se le llama *relaciones*. Los **atributos** son los conjuntos de las entidades así se describen en las bases de datos y las relaciones eran asociaciones entre varias entidades. Los diagramas de E-R es por lo general es una estructura lógica de la base de datos, los componentes son: rectángulos representan conjuntos de entidades, elipses representan los atributos, rombos representa las relaciones entre las entidades, lineas es la unión entre las entidades y los atributos además de las relaciones y la entidades[22].

A mediados de los 70´s fue presentado el Modelo Entidad- Interrelación, propuesto por Peter Chen para visualizar y presentar de manera conceptual los problemas de forma global. Es un modelo que al pasar el tiempo ha experimentado muchas modificaciones y generando un elemento mas útil para la resolución de problemas. El modelo Entidad-Interrelación se denota con grafos y tablas, mediante el conjunto de símbolos haciendo que el sistema las relaciones y haciendo un pseudocodigo de manera más sencilla [26].

2.5.1. Entidades

Una entidad es lo que describe a una tabla, es decir, el nombre con que se da referencia a una tabla, por ejemplo, en un ambiente de biblioteca tenemos como entidades Libro, Editorial y Genero por mencionar algunas. [5]. Ver representación en la Figura 2.6.

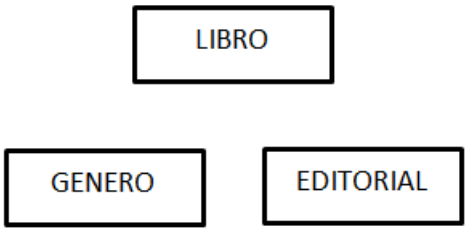


Figura 2.6: Representación de entidades

Las entidades se dividen en dos clases, las regulares y las débiles las cuales se describen como sigue:

- **Regulares:** estas existen por sí mismas.
- **Débiles:** esta depende de la entidad regular, ya que si se elimina la entidad regular desaparece la entidad débil. Este tipo de entidades se representa por dos rectángulos con su nombre en el interior [5].

2.5.2. Interrelación

Esta se define como la asociación de correspondencias entre entidades [5].

Es la unión de dos o más conjuntos de entidades existentes, ademas de ser la vinculación de otras entidades, por ejemplo, ver Figura 2.7: la entidad ALUMNO se interrelacionan con la entidad CURSO, con la leyenda Asistir, o dicho de otra forma "Luis " asistió al curso de "Redes neuronales". Este tipo de interrelación se representa con rombo etiquetando el nombre de la interrelación, conectando a las dos entidades.

Figura 2.7: Ejemplo de interrelación

Se caracteriza por:

- Nombre: etiqueta del rombo
- Grado: son las entidades que participan en la interrelación, estas pueden ser de grado 2 o bien binarias, de grado 3 o terciarias cuando son tres entidades, o grado n.
- Tipo de correspondencia: Es cuando la correspondencia puede presentarse de 1:1, es decir que la interrelación aparece como máximo una ocurrencia para cada una de la entidades, 1:N cuando una de las entidades puede tener un número indefinido de ocurrencias, y N:M cuando existe un número mayor de uno para una de las entidades.

2.5.2.1. Atributos

Son las características de las entidades, cada uno de los atributos deberá nombrarse con la finalidad de recordar el contenido de la entidad a la cual esta asociada. Por ejemplo, la entidad LIBRO tiene como atributos a Titulo, Año, Paginas, etc., estos atributos son colocados de manera que se entienda, de que se esta hablando[20].

En el modelo de Chen, los atributos son representado con óvalos, estos contiene el atributo de las entidades.

2.5.2.2. Dominio

Los valores que pudieran tener los atributos, teniendo una existencia propia independiente de la entidad. El dominio es representado por óvalos y en el interior guarda su nombre [5].

2.5.2.3. Tabla o esquemas

Son los conjuntos de entidades o también llamados grupos de entidades. Una tabla está compuesta por filas y columnas.

- Cada fila representa los datos almacenados de cada uno de los atributos de la relación, también llamado **tupla.**
- Cada columna de la entidad contenida en una tabla representa los **atributos** y cada una de las columnas debe contener un nombre único .

Todos los valores de una tabla deben sujetarse a los formatos de datos y cada uno de los atributos debe identificarse de forma única con cada una de las filas y cada intersección de fila y columna su valor debe ser único para representarlo.

Cada atributo contiene valores determinados como dominios de atributos [30].

2.5.2.4. Clave primaria

Una clave primaria es una columna o combinación de columnas dentro de una tabla cuyo valor identifica unicamente a cada fila de la tabla. Cada tabla tiene una única clave primaria [14].

Son los atributos de una tabla que a su vez determinan a otros pero estas solo pueden ser únicas para cada tabla, por ejemplo en la tabla ALUMNO la clave primaria es el Numero_Cuenta, como se muestra en la Figura 2.8.

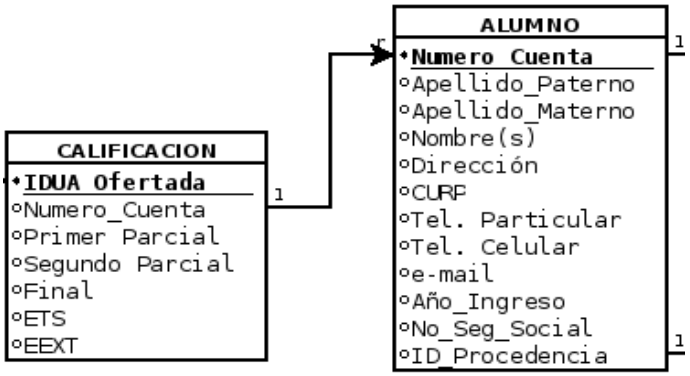


Figura 2.8: Representación de clave primaria

"Una clave se compone de uno o más atributos que a su vez determinan otro" , se pueden definir varias claves como superclave, "que es cualquier clave que identifica cada entidad de manera única" [20].

Clave candidato se puede describir como una superclave sin redundancias [20].

Clave secundaria son aquellas que se utilizan para recuperación de datos, "para que una clave secundaria sea efectiva se tiene que hace runa búsqueda especifica" [20].

2.5.2.5. Cardinalidad

La cardinalidad se define como el máximo o mínimo de las entidades participantes, su representación gráfica se expresa con número específicos asociados a las entidades relacionadas con el rombo que representa la relación [5].

2.5.2.6. Relación uno a muchos

Para explicar las relaciones se usan los conjuntos, determinemos el conjunto A y B, que en BD serán las entidades y la relación es de la siguiente forma:

La entidad A se relaciona con cualquier entidad de B, estas pueden ser desde una o más entidades, pero sólo una entidad de B puede relacionarse con A. En BD la expresión se denota 1:M [25].

2.5.2.7. Relación uno a uno

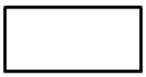
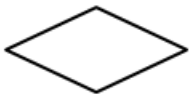
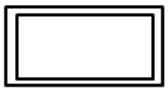
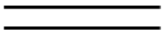
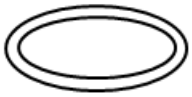
La entidad A únicamente puede relacionarse con una entidad de B y viceversa, en BD se expresa de la siguiente forma 1:1 [25].

2.5.2.8. Relación muchos a muchos

La entidad A se relaciona con cualquier entidad de B, con más de una de la entidad B y la entidad B se relaciona con cualquier tipo de entidad de A, con más de una. También se expresa como M:N [25].

2.5.3. Diagrama Entidad - Relación

Son diagramas de ER una breve descripción de una BD, esto para ver de manera más sencilla la BD. Este diagrama tiene los siguientes elementos ver el Tabla2.1, describiendo el símbolo y el nombre de cada uno de los elementos que lo conforman.

NOMBRE	SIMBOLO	DESCRIPCIÓN
Rectángulo		Entidad
Ovalo		Atributo
Rombo		Relación
Línea		Unión de atributos a relación
Rectángulos dobles		Entidades débiles
Líneas dobles		Participación total
Elipses discontinuas		Atributo derivado
Elipses dobles		Atributo multivalorado

Cuadro 2.1: Símbolos E-R [27].

En la Figura 2.9 se muestra un ejemplo de diagrama de entidad relación, que consta de de tres entidades: cine, genero, programación, todas ellas relacionadas a través de la relación pertenece y tiene.

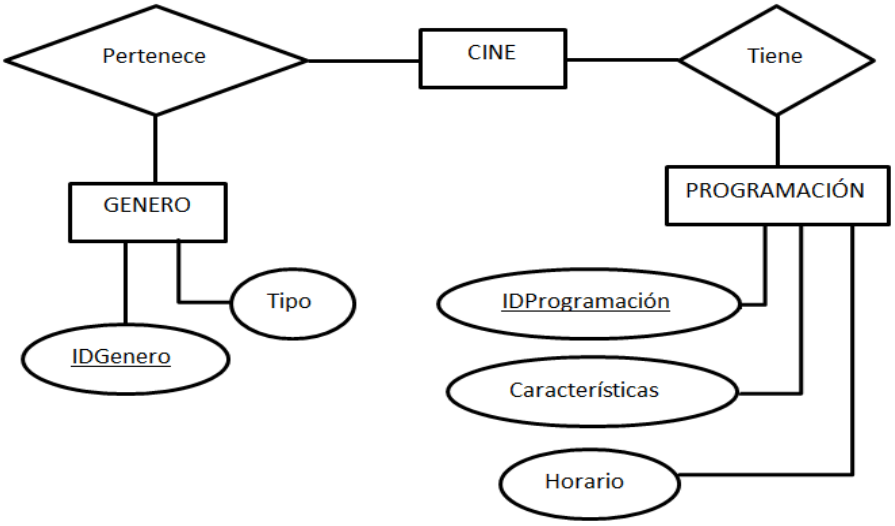


Figura 2.9: Ejemplo de un Diagrama E- R

2.5.3.1. Conjunto de entidades débiles

Este conjunto es caracterizado por no tener lo suficientes atributos para poder ser considerada una clave primaria, para que este conjunto de entidades débiles sean considerada tendrá que ser parte de entidades identificadoras o propietarias, en otras palabras la entidades débiles dependen de las entidades identificadoras.

2.6. Álgebra relacional

En 1970 el científico informático Edgar Frank Codd, implemento una serie de operaciones para el modelo relacional mismas que fueron implementadas y denominadas como álgebra relacional. Esta implementación es parte de la teoría de conjuntos, demostrando que cualquier acceso a una BD se puede realizar con estos operadores [3].

El álgebra relacional define la forma en como se manipulan los datos dentro de las tablas, a esto se le denomina operadores, la manera en como los operadores actúan sobre las tablas es mediante la creación de nuevas relaciones que permitan concentrar datos específicos que el usuario requiera obtener. Las nuevas relaciones no crean nuevas tablas, solo se crean listas a partir de la combinación de columnas y reglones que se desean [29].

El álgebra relacional se puede considerar como un lenguaje objetivo, por lo que puede convertir en un comando de consultas.

2.6.1. Operaciones relacionales básicas

Estas operaciones básicas son las siguientes: SELECT, PROJECT, JOIN, INTERSECT, UNION, DIFFERENCE, PRODUCT, DIVIDE, para definir estos operadores no es necesario definir muchas cosas únicamente es necesario ilustrar como funciona cada operador.

2.6.1.1. SELECT

Muestra valores de filas o registros que contienen las tablas, opera de manera horizontal. En la Figura 2.10 se muestra un ejemplo de SELECT [29].

Tabla original

ID	Nombre	Edad
1	Ernesto	15
2	Alberto	14
3	Alma	15
4	Carlos	14
5	Maria	14

SELECT : escoge renglon donde Edad = 15

ID	Nombre	Edad
1	Ernesto	15
3	Alma	15

Figura 2.10: Ejemplo de sentencia SELECT

2.6.1.2. PROJECT

Proporciona valores de atributos obteniendo nuevos ficheros a partir de tablas existentes, es decir obtiene y elije valores de columnas tablas existes. En la Figura 2.11 se muestra un ejemplo de como se efectuá la proyección [29].

Tabla original

ID	Nombre	Edad	Cuidad
1	Ernesto	15	Puebla
2	Alberto	14	Monterrey
3	Alma	15	D.F.
4	Carlos	14	Edo. México
5	Maria	14	Edo. México

PROJECT
Proyecta de la tabla original las columnas Nombre y Ciudad

Nombre	Ciudad
Ernesto	Puebla
Alberto	Monterrey
Alma	D.F.
Carlos	Edo. México
Maria	Edo. México

Figura 2.11: Ejemplo de sentencia PROJECT

2.6.1.3. PRODUCT

Este operador similar a una multiplicación, por ejemplo si una tabla tiene 3 filas y la otra tabla 2 filas, se realiza una operación de multiplicación 3 X 2 = 6, a esto también se le denomina producto cartesiano [20].

PRODUCTO

A	B	C
a1	b1	c1
a2	b2	c2
a3	b3	ce

D	E
d1	e1
d2	e2

RESULTADO

A	B	C	D	E
a1	b1	c1	d1	e1
a1	b1	c1	d2	e2
a2	b2	c2	d1	e1
a2	b2	c2	d2	e2
a3	b3	ce	d1	e1
a3	b3	ce	d2	e2

s

Figura 2.12: Ejemplo de sentencia PRODUCT

2.6.1.4. JOIN (o reunión natural)

Este operador combina horizontalmente dos tablas a partir de valores comunes, estos valores deben tener valores en común. El resultado de join se efectuara en varias etapas y para esto se utilizarán las tablas de la Figura 2.13. Ahora se describen las etapas:

AUTO

ID_Auto	Marca	Modelo
1	Nissan	2002
2	Chevrolet	2005
3	Honda	2006
4	Ford	2007

SERVICIO

Id_Serv	ID_Auto	Fecha_Servcio	Tipo_Servicio
11	1	20/11/2008	cambio de aceite
12	2	02/03/2013	revision de frenos
13	4	05/09/2012	cambio de balatas

Figura 2.13: Ejemplo de sentencia JOIN

1. Se realiza operación PRODUCT a las tablas de la Figura 2.13 que se requiera combinar y el resultado es el que se muestra en la Figura 2.14.

PASO 1 PRODUCT

ID_Auto	Marca	Modelo	Id_Serv	ID_Auto	Fecha_Servcio	Tipo_Servicio
1	Nissan	2002	11	1	20/11/2008	cambio de aceite
1	Nissan	2002	12	2	02/03/2013	revision de frenos
1	Nissan	2002	13	4	05/09/2012	cambio de balatas
2	Chevrolet	2005	11	1	20/11/2008	cambio de aceite
2	Chevrolet	2005	12	2	02/03/2013	revision de frenos
2	Chevrolet	2005	13	4	05/09/2012	cambio de balatas
3	Honda	2006	11	1	20/11/2008	cambio de aceite
3	Honda	2006	12	2	02/03/2013	revision de frenos
3	Honda	2006	13	4	05/09/2012	cambio de balatas
4	Ford	2007	11	1	20/11/2008	cambio de aceite
4	Ford	2007	12	2	02/03/2013	revision de frenos
4	Ford	2007	13	4	05/09/2012	cambio de balatas

Figura 2.14: Ejemplo de sentencia Paso 1 PRODUCT

2. Después de realiza SELECT al atributo de la tabla resultante donde los valores son iguales, a estos valores se les denomina como columnas unidas. En la Figura 2.15 se muestra el resultado

de este paso.

ID_Auto	Marca	Modelo	Id_Serv	ID_Auto	Fecha_Servcio	Tipo_Servicio
1	Nissan	2002	11	1	20/11/2008	cambio de aceite
2	Chevrolet	2005	12	2	02/03/2013	revision de frenos
4	Ford	2007	13	4	05/09/2012	cambio de balatas

Figura 2.15: Ejemplo de sentencia Paso 2 SELECT

3. Por último se realiza operación PROJECT, para eliminar los columnas repetidas de la tabla anterior, la Figura 2.16 muestra el resultado de esta último paso.

PASO 3
Eliminar columnas repetidas

ID_Auto	Marca	Modelo	Id_Serv	ID_Auto	Fecha_Servcio	Tipo_Servicio
1	Nissan	2002	11	1	20/11/2008	cambio de aceite
2	Chevrolet	2005	12	2	02/03/2013	revision de frenos
4	Ford	2007	13	4	05/09/2012	cambio de balatas

Figura 2.16: Ejemplo de sentencia Paso 3 PROJECT

Características principales de Join natural:

- No incluye filas cuyos datos no coincidan.
- A los atributos o columnas que se les aplica JOIN solo puede aparece una sola vez.

El operador JOIN cuenta con otros operadores como equiJOIN, outerJOIN, left outer JOIN, right outer JOIN que enseguida se describen:

- equiJOIN: el nombre de equiJOIN lo toma de la igualdad (es decir =), este operador une dos tablas con condición de igualdad, comparando tablas especificas y el resultado de esta operación no elimina columnas repetidas.
- outerJOIN: conserva pares iguales y los desiguales los deja nulos en otra tabla.
- left outer JOIN proporciona filas de la tabla 1 (por dar un ejemplo) que sus datos no sean iguales a los de tabla 2 .
- right outer JOIN obtiene filas de la tabla 2 incluyendo filas que no tienen valores iguales a la tabla 1 [20].

2.6.1.5. UNIÓN

Combina e intercala filas, para esto las columnas o atributos deben ser iguales.

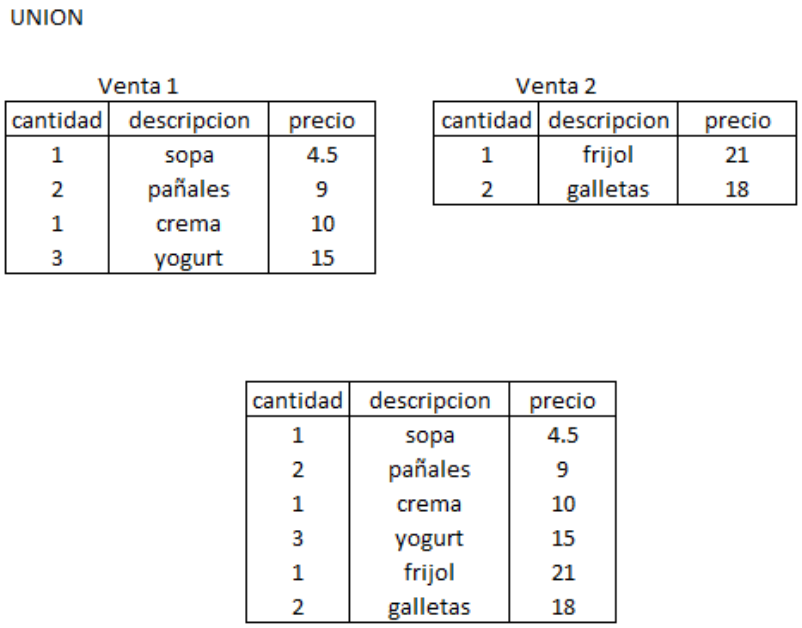


Figura 2.17: Ejemplo de sentencia UNION

2.6.1.6. INTERSECT

Las tablas deben de ser compatibles con los valores que se almacenan en las tablas, es decir si los tipos de datos son diferentes no se puede realizar la operación, en la Figura 2.18 se muestra un ejemplo de esta sentencia [29].

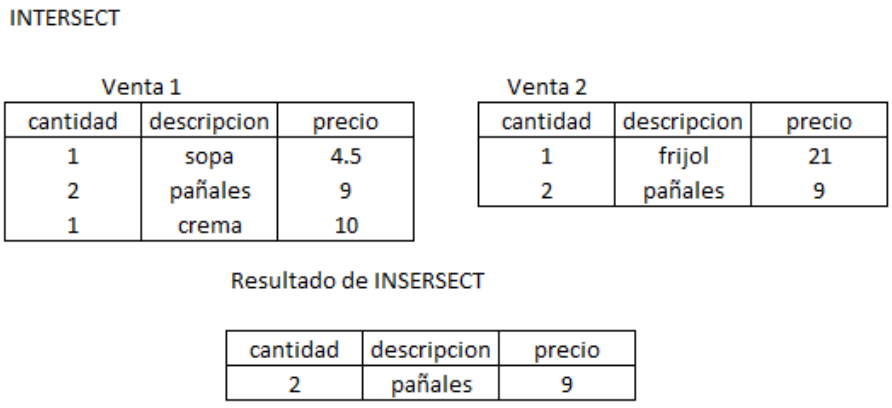


Figura 2.18: Ejemplo de sentencia INTERSECT

2.6.1.7. DIFFERENCE

La operación DIFFERENCE es similar a una resta, es decir quita registros de una tabla y de otra, mostrando unicamente los datos restantes de las tablas, en la Figura 2.19 se muestra como es una operación DIFFERENCE [29].

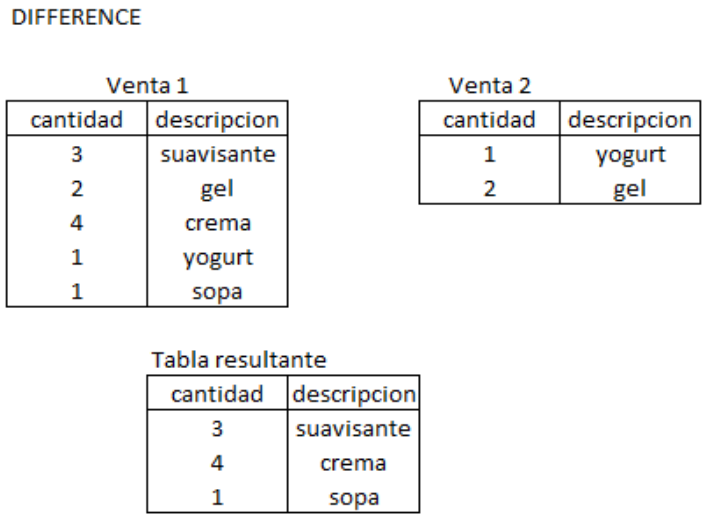


Figura 2.19: Ejemplo de sentencia DIFFERENCE

2.7. Dependencias funcionales

Los atributos y las dependencias están formados por relaciones, la diferencia entre ellos es que las dependencias son más complicadas de encontrar, ya que los atributos los elige el diseñador. Las dependencias se dan entre atributos y subconjunto de atributos y son consecuencia de la estructura que el mundo real ofrece

Otra manera de explicar las Dependencias funcionales son restricciones de dos o más columnas de una tabla en la BD, estas restricciones pueden clasificarse en valor-base *versus* valor-neutral. Estas restricciones involucran comparaciones de columna, el valor-base emplea operadores de comparación tales como $<$,$=$, o $>$, las restricciones de llave primaria y llave foránea son restricciones de valor-neutral [18].

La definición de una dependencia funcional es *Si R e un esquema de relación y A y B son conjuntos de atributos no vacíos en R, se dice que B es funcionalmente dependiente en A si y sólo si cada valor de A en R tiene asociado exactamente un valor en B en R [24].*

2.7.1. Superclaves, claves candidatas y claves primarias

Superclave: identifica de manera única al o los atributos de cada tabla, en otras palabras se utiliza para distinguir una fila de otra.

Clave candidata: Es un identificador mínimo, estas claves no deben pertenece a un subconjunto de atributos.

Clave primaria: es una clave candidata que se usa para identificar la columnas en una relación. La mejor opción para determinar que una clave candidata pueda ser usada como clave primaria es cual representa en realidad a cada tabla.

2.8. Normalización de bases de datos

La normalización es un método más de la parte del diseño de la base de datos, el objetivo es eliminar la redundancia de los datos que existe en los esquemas de la relaciones entre las entidades, también puede permitir la recuperación de la información de manera más eficiente y con facilidad.

Para el diseño de correcto de bases de datos se requiere de conocer y comprender las relaciones que existen entre los datos, y saber expresar dichas relaciones. Una vez que se ha diseñado un modelo entidad-relación, es necesario aplicar una serie de reglas a las relaciones, para obtener un modelo relacional correcto. A dicho proceso se le conoce con el nombre de normalización de bases de datos. De una manera simple y concisa, el proceso de normalización consiste en convertir una relación que tiene problemas, en varias relaciones que no los presentan [12].

Aunque no es obligatorio normalizar una base de datos, sí es muy recomendable realizar este proceso, para evitar problemas que se presentan en etapas avanzadas del desarrollo de sistemas software que usan bases de datos. Algunos de estos problemas son la duplicación de datos, anomalías en las consultas realizadas y un bajo rendimiento. Por lo tanto, las principales razones por las que conviene normalizar una base de datos son las siguientes [2]:

1. Se elimina la redundancia de datos.
2. Se reducen los problemas de actualización de los datos en las tablas, se protege la integridad de los datos.
3. Se evita que no puedan almacenarse en la base de datos ciertos tipos de datos.

Una base de datos se dice que se encuentra normalizada si cumple con tres requisitos naturales o reglas, a las que se les conoce como "las tres formas normales".

2.8.1. Primera Forma Normal (1FN)

El modelo entidad-relación acepta que las entidades tengan sub-estructuras [28]. Además, permite que los atributos sean compuestos. Por ejemplo, se permitiría que un campo llamado teléfono, acepte en un mismo registro los números de teléfono celular, fijo de casa y fijo de trabajo. Estos valores no serían atómicos. Un dominio es atómico, *"si se considera que los elementos en ese dominio son unidades indivisibles"* [28].

Una relación está en su primera forma normal, o 1FN, si sus atributos contienen valores atómicos.

En caso de que una tabla no cumpla con la 1FN, la solución general es separar el atributo que viola esta forma.

Por ejemplo, suponer que en una tabla se tienen los campos siguientes (ver Figura 2.20):

- Id
- Nombre
- Teléfono

ID	NOMBRE	TELÉFONO
1	Lupe	56 78 90 10
2	Juan	55 78 34 21
3	Beto	67 90 78 32
4	Pedro	10 78 42 56
5	Alma	56 43 23 11

Figura 2.20: Datos generales sin normalizar

Como el teléfono puede ser celular, trabajo, casa, recados u otro, entonces una tabla con este atributo no se encuentra en 1FN.

Para lograr que esta tabla cumpla con 1FN, se puede separar el atributo teléfono, dando lugar a otra tabla, ver Figura 2.21.

Tabla 1

- Id
- Nombre

Tabla 2

- Id
- Teléfono

ID	NOMBRE	TELÉFONO
1	Lupe	56 78 90 10
2	Juan	55 78 34 21
3	Beto	67 90 78 32
4	Pedro	10 78 42 56
5	Alma	56 43 23 11

ID	NOMBRE
1	Lupe
2	Juan
3	Beto
4	Pedro
5	Alma

ID	TELÉFONO
1	56 78 90 10
2	55 78 34 21
3	67 90 78 32
4	10 78 42 56
5	56 43 23 11

Figura 2.21: Ordenado a Primera Forma Normal

De esta forma, todos los atributos cumplen con ser atómicos.

2.8.2. Segunda Forma Normal (2FN)

Para que una relación se encuentre en la segunda forma normal, o 2FN, el primer requisito que debe de cumplir es el de ser 1FN.

La 2FN se aplica las relaciones que con claves primarias compuestas por más de un atributo. Por lo tanto, si una relación cumple con 1FN, y su clave primaria es no compuesta (es simple), entonces ya se encuentra en 2FN. Si la clave primaria es compuesta, entonces hay que verificar la regla de 2FN.

El requisito segundo que se debe de cumplir en una relación para estar en 2FN, es que todos sus atributos que no forman parte de la clave principal, tienen una dependencia funcional completa respecto de todas las claves existentes en el esquema. Es decir, para determinar cada atributo que no es clave primaria, se necesita la clave primaria completa.

2.8.3. Tercera Forma Normal (3FN)

Un requisito para que una relación se encuentre en 3FN, es que debe cumplir con 2FN. La regla que impone 3FN es que cada atributo que no está incluido en la clave primaria no debe de depender transitiva mente de la clave primaria. En otras palabras, lo que se debe de cumplir es que ninguna columna de la relación pueda depender de una columna que no tenga una clave, y que no puedan haber datos derivados en la relación.

2.9. Lenguaje estructurado de consulta SQL

El SQL (*Structured Query Languaje*), anteriormente nombrado SEQUEL (*Strutured English Query Language*), es un lenguaje importante para las bases de datos del calculo relacional, se implementó

en un proyecto llamado SEQUEL-XRM. En San José California, IBM desarrolló en sus laboratorios el prototipo de SQL [28]. Este prototipo se puede utilizar como un lenguaje de definición de datos o como manejo de datos. Este lenguaje fue desarrollado también con el propósito de poner a prueba el modelo relacional de Codd, el lenguaje puede generar reportes, datos actualizados dentro del modelo E-R.

Ventajas que aportan :

- Se emplea en la mayoría de los sistemas actuales.
- El uso de operadores permite realizar operaciones en minutos.

En un concurso donde participaron empresas reconocidas como Oracle y Sybase, empezaron a entregar sus productos en SQL, esto provocó que SQL fuera más comercial. ANSI adoptó a SQL como estándar en 1986, un año más tarde se transformará en una estándar ISO, a partir de este momento SQL es el nombre y el año, posteriormente han surgido diversas versiones ejemplo SQL/89 y la versión actual es SQL/92.

SQL proporciona algunas funciones, que a continuación se mencionan:

- Definición de datos: los usuarios pueden modificar y definir la estructura en la que se almacenaran y relacionan los datos.
- Recuperación de datos: los programas y usuarios pueden recuperar los datos que están almacenados en la BD.
- Manipulación de datos: permite a los usuarios y programas actualizar datos que se encuentran almacenados en las BD, añadiendo y eliminando nuevos datos.
- Control de acceso: este permite proteger a los datos que están almacenados, se utilizado para restringir permisos a un usuario no autorizados, evitando que se realicen modificaciones y añadir datos,
- Comparación de datos: se usa para compartir datos.
- Integridad de datos: este declara la integridad de la BD, cuidando que no se dañen los datos en caso de actualización o fallo inesperados ocasionados por el sistema [14].

2.9.1. Consultas estructurales

En este apartado se concentra en la estructura que lleva una BD con código SQL, ahora mencionan algunas de las consultas que son necesarias para crear una BD.

- para empezar a realizar una BD se necesitamos crear la BD, para esto utilizamos la siguiente consulta
  CREATE DATABASE;
  la sintaxis es la siguiente *CREATE DATABASE "nombre_de_la_BD"* y finalizando con punto y coma.

La nueva BD contendrá las tablas creadas para almacenar los datos.

- Para usar la BD creada se utiliza la consulta *USE*, ejemplo:
  USE NOMBRE_BASE_DATOS;

- Puede darse el caso de que te requiera eliminar una BD, para esto se utiliza la consulta *DROP* especificando cual es la BD que se desea eliminar y colocando al final de la consulta el punto y coma. Esta consulta tiene un desventaja muy importante resaltar, tiene el problema que si eliminas una BD eliminas todo lo que contenga la BD incluyendo tablas, no queda nada y este proceso es irreversible.
  DROP DATABASE "NOMBRE_BASE_DATOS;"

- Conociendo ya estas primeras consultas para la creación de las BD, después de esto se crean las tablas con la consulta *CREATE TABLE* que a continuación se mostrará con pequeño ejemplo: Definimos de una tabla Libro los siguientes atributos IdLibro, titulo, autor, año, editorial y edicion. utilizaremos la consulta *CREATE TABLE* seguida del nombre de la tala, después abrimos paréntesis para agregar los atributos que tendrá la tabla separados por una coma (las coma son obligatorias por la sintaxis) y salto de línea (es opcional, ya que es mas sencillo leer el código con saltos de línea), seguido del nombre del campo y su tipo de dato,este ultimo se vera en el siguiente apartado.

```
CREATE TABLE Libro(
IdLibro int(4),
titulo varchar(50),
autor varchar(50),
año int(4),
editorial(20),
edicion(20),
PRIMARY KEY (IdLibro));
```

De esta manera se crea una BD y sus tablas. Los tipos de datos son muchos pero no todos son empleados, esto es dependiendo del contenido que se almacenara en las tablas [11].

2.9.2. Tipos de datos

Los tipos de datos son variados, pero no no todos se emplean esto es dependiendo del contenido de las tablas a emplear sera los tipos de datos usuados. Los tipos de datos siempre se escriben en mayúsculas, ahora se en listan algunos:

- **TINYINT:** Enteros desde 0 a 255 sin signo o de -128 a 127 con signo.
- **SMALLINT:** Enteros de 0 a 65,535sin signo, o de -32,768 a 32-767 con signo.
- **MEDIUMINT:** Entero de 0 a 16,777,215 sin signo o de -8,388,608 a 8,388,607 con signo.
- **BIGINT:** Entero de 0 a 18,446,744,073,709,551,616 sin signo o desde el negativo -9.223.372.036,854,775,808 a 9,223,372,036,854,775,807 con signo.

- **FLOAT (M,D):**Son números flotantes y la precisión es de n dígitos. M es la cantidad máxima de dígitos, sin tomar en cuenta el signo y punto decimal y D es el dígito en decimal pueden ser mas de un decimal.
- **DOUBLEO REAL (M,D):** Número en coma flotante de doble precisión, M yD son el caso de FLOAT. El rango es de 2,225073E-308 y -1,7976931E308.
- **DATE:** es la fecha contiene el año a cuatro cifras, dos cifras el mes y dos cifras el día o bien dos cifras para cada uno y con el siguiente formato AAAA-MM-DD o de la siguiente manera AA-MM-DD.
- **DATETIME:** es como el caso anterior pero ahora se agrega la hora y el formato es el siguiente AAAA-MM-DD HH:MM:SS.
- **TIME:** Contiene la hora del día, representado en horas, minutos y segundos, también se puede especificar el uso horario utilizando **time with timezone**, y los formatos son los siguientes HH:MM:SS o HHMMSS o bien HHMM.
- **YEAR:** unicamente se almacena el año en formato AAAA
- **CHAR** (n) son caracteres de longitud fija, n significa que el usuario puede colocar la cantidad deseada.
- **VARCHAR** (n): Son cadenas de caracteres, con longitud variable máxima de n, esto especificado por el usuario.
- **TINYTEXT:** son para agregar textos en formato plano con un máximo de caracteres de 255.
- **TEXT:** es muy parecido al anterior pero la longitud es mayor, con un máximo de 65,535 caracteres.
- **MEDIUMTEXT:** de manera similar a los dos anteriores con texto plano pero con máximo de 16,777,215 caracteres.
- **LONGTEXT:** texto plano con un máximo de 4,294,967,295 caracteres.
- **TINYBLOB:** Son archivos binarios (puede ser texto RTF, imágenes, ect.) con un máximo de 255 bytes.
- **BLOB:** archivo binario con valor de BLO normal y rango de 0 a 65,535 bytes.
- **MEDIUMBLOB:** archivo binario de valor BLOB medio con rango de 0 a 2^{24} -1.
- **LONGBLOB:** archivo binario de valor BLOB grande con rango de 0 a 2^{32} -1.
- **INT O INTEGER:** Son enteros, dependientes de la máquina.
- **NUMERIC O DECIMAL:** (p,d): Esta formado por p dígito, d pertenece a la parte decimal. Ejemplo numeric (2,1) [14], [11].

Ademas de utilizar tipos de datos también se utilizan operadores lógicos y aritméticos, esto cuando se realizan consultas, en el Tabla 2.2 se muestran los operadores que utiliza SQL.

OPERADORES ARITMETICOS	
OPERADOR	DESCRIPCIÓN
>	Menor que
<	Mayor que
=	Comparación de igualdad o igual a
<=	Mayor o igual
=>	Menor o igual
< > o ! =	Desigualdad o distinto de
LIKE	Parecido a
BETWEEN	Comprendido entre dos valores
IN	Consultas en tablas distintas.
OPERADORES LÓGICOS	
AND	Une dos coincidencias, ambas deben ser iguales.
OR	Une dos coincidencias y al menos una de ellas coincida.
NOT	Elige dos condiciones que no cumplan con la condición.

Cuadro 2.2: Operadores aritméticos y lógicos [14]

SQL permite el uso de valores como nulos, no nulos, etc., también permite al usuario determinar qué valores son nulos o pueden contener valores nulos en los atributos. En el Tabla se mostraran los posibles valores que el usuario determine para cada atributo del Tabla 2.3.

VALOR	DESCRIPCIÓN
NULL	No tiene valor cuando se crea o cuando se modifique un valor.
NOT NULL	Indica que esta columna no puede estar vacía, debe contener un valor, si no dará errores.
DEFAULT	Se indica el valor que se almacenará por default en cada campo, si los valores son alfanuméricos se colocan comillas simples y si son numéricos se colocan sin comillas.
ZEROFILL	Este valor se agrega a tipos de valores numéricos, esto es cuando se requiere que se llene con ceros los dígitos de más.
UNIQUE	Este se utiliza para que no se repitan los valores en diferentes registros
UNSIGNED	Este valor impide que se almacenen valores negativos.
AUTO_INCREMENT	Este valor incrementa de manera automática el valor en casos del tipo de dato numérico.
PRIMARY KEY	Este define un campo como clave primaria, permitiendo que no haya valores repetidos.

Cuadro 2.3: Valores para los campos

SQL permite el uso de llaves foráneas, estas claves se determinan a campos que hagan relación con otra tabla y esta a su vez deben de ser llaves primarias. Así mismo existen tres clausulas en el query cuando se crea una tabla como son *REFERENCES, ON UPDATE, ON DELETE* como se describen a continuación:

- *REFERENCES:* indica qué tabla es clave principal de la tabla foránea.
- *ON UPDATE:* indica que se debe actualizar
- *ON DELETE:* indica que se debe de hacer si se elimina algún valor de un campo, esta última clausula tiene tres posibilidades
  - *CASCADE.* Actualiza o elimina registros.
  - *SET NULL.* Establece valores *NULL,* si el registro es eliminado.

- *RESTRICT.* Imposibilita la eliminación y actualización de campos.

La creación de índices son utilizados para realizar los procesos de búsqueda son más rápido utilizando la consulta *CREATE INDEX*, como regla general para crear un índice la sintaxis es la siguiente:

CREATE INDEX índice ON table (campo);

La eliminación de indices se utiliza *DROP INDEX,* esto es muy útil por que después de eliminar las tablas, los índices pueden generarse sin ningún problema, y para esto utilizaremos la siguiente consulta:

DROP INDEX índice;

Los requerimientos de los clientes van modificando de acuerdo a sus necesidades, es por que se requiere de modificaciones pero sin perder los datos que ya se tienen, todo nos indica que la tabla puede ser modificada y para eso existe la siguiente consulta: *ALTER TABLE.*
Para hacer esto es importante conocer estas consulta, que no son necesarias utilizarlas todas, unicamente las que se necesiten e inclusive pueden usarse mas de una vez.

1. Debe existeir en la BD, identificamos la tabla que vamos a modificar.
2. Añadimos campos nuevos con la consulta *ADD.*
3. Utilizando *CHANGE* cambiamos los valores de la tabla a modificar.
4. Y por últimо se eliminan campos con la consulta *DROP.*

ALTER TABLE tabla
ADD campo definiconDelCampo,
CHANGE campAntiguo campoNuevo definicionDelCampo,
DROP campo;

Para añadir registros en una tabla se utiliza la consulta INSERT INTO la sintaxis es la siguiente:

INSER INTO tabla (camp1, camp2, camp3,..., campoN)
VALUES (valor1, valor2, valor3,..., valorN);
Reglas que se deben cumplir para el llenado de los datos:

1. Los valores deben de llenarse tal cual aparecen en el orden en el que se crearon.
2. Los números se colocan sin comillas
3. Cuando son cadenas de caracteres, se colocan con collas dobles.
4. La consulta se finaliza con punto y coma.

La consulta SELECT para encontrar datos dentro de las tablas, con esta consulta encontraremos datos de campos especificados, y esta es la sintaxis:

SELECT campo1, campo2, . . . ,campoN *FROM* tabla *WHERE* condición;
Ahora para supongamos que queremos todos los datos que tengan una condición, unicamente se coloca un asterisco (*) y la sintaxis es la siguiente:

*SELECT * FROM* tabla *WHERE* condición = "algo";

Ahora si lo que se requiere es recuperar todos dato o alguno de ellos utilizamos las siguientes consultas:

*SELECT * FROM* tabla;

SELECT campo1, campo2 *FROM* tabla;

Es importante conocer más cláusulas además de *WHERE*, para esto se en listan otras cláusulas que pueden utilizadas:

- GROUP BY: Agrupa datos y registros para generar informes de los datos de las tablas, esta es la sintaxis

  SELECT campo1, campo2, *FROM* tabla *GROUP BY* condición;

- SUM ... AS: permite sumar valores numéricos para explicarte esta cláusula se tomara un ejemplo del libro [11], que nos ayudará a entender esta clausula

  SELECT nombre *SUM* (salario) *AS* totalSueldos *FROM* empleados *GROUP BY* departamento;

- AVG ... AS: con esta cláusula se calcula el promedio, el término proviene de la palabra inglesa *AVERAGE*, la sintaxis es la siguiente.
  SELECT nombre *AVG* (salario) *AS* media *FROM* empleados *GROUP BY* departamento;

- ORDER BY: se puede obtener datos de tablas mediante esta consulta de manera ordenada, el orden puede ser ascente, es decir, de menor a mayor o descendente de mayor a menor y los modificadores son *ASC* y *DESC* respectivamente.
  SELECT campo1, campo2 *FROM* tabla *ORDER BY* campo2 *DESC*;

- COUNT ... AS: esta consulta muestra cuántos registros hay en un grupo y cuántos cumplen con determinadas condiciones.
  SELECT COUNT (*) AS campo1 *FROM* tabla1, *ORDER BY* tabla2;

- LIKE: esta consulta sirve para encontrar datos que son similares, en la sintaxis se coloca un asterisco, esto indica que cualquier carácter o conjunto de caracteres que contenga o empiece con la letra "X".
  SELECT campo1, campo2 *FROM* tabla1 *WHERE* campo3 *LIKE* "X * ";

- LIMIT: limita la cantidad de registros permitidos en una consulta, es decir podemos solo consultar los 15 primeros y así sucesivamente hasta terminar con todos los registos, esto se hace de esta manera para que la cantidad de datos no sea erróneo e incomodo de trabajar.
  *SELECT * FROM* trabajos *WHERE* campo1= "XY" *LIMIT* 15;

2.10. Sistema Gestores de Bases de Datos (SGBD)

Se ha visto como se convierten los modelos en ficheros datos, pero desafortunadamente tienen muchas limitaciones, es por eso que los ficheros ahora no son sofisticados, por eso crearon los Sistemas Gestores de Base de Datos, que aportan una interfaz para conectar cualquier programa.

Los Sistemas Gestores de Base de Datos (SGBD), o DBMS por sus siglas en inglés *Data Base Management System,* son el software que facilita el establecimiento y utilización de una BD a los usuarios, mediante los diversos programas que existen para el diseño de la BD, permitiendo procesar, describir, administrar y recuperar datos. Los SGBD cargan datos almacenados por el usuario en unidades de disco u otros dispositivos para el acceso de los datos. El SGBD se adapta a los requerimientos del sistema que procesará los datos, permitiendo que se puedan realizar las operaciones sobre los ficheros, los registros y los indices de la base de datos.

Funciones de los SGBD:

- Definición: Permite al diseñador definir los elementos de su estructura, la relación que existe entre ellas, características físicas y lógicas.

- Manipulación: Esto se da después de definir los datos a manera de consulta de esta manera los usuarios pueden actualizar y recuperar los datos, las consultas son de dos tipos: consulta selectiva y consulta sobre el total de los datos. La actualización se puede realiza con tres operaciones: borrado o eliminación, modificado y inserción.

- Control: Facilita la tarea del administrador permitiendo funciones como cambiando la capacidad de los registros o ficheros, estadísticas y seguridad [2].

Características

- Diseño de datos

- Diseño físico

- Control de redundancia e inconsistencia:

- Seguridad: Se puede limitar el acceso a los usuarios en algunas partes de la BD, para determinar que hacer cada usuario.

- Almacenamiento: el SGBD se encarga de guardar cada byte, el administrador solo define el modelo y lo trabaja desde el nivel superior [6].

La implementación y crecimiento de las bases de datos, provoca que al gestionar los datos sea más complicado, un sistema gestor de base de datos forma parte de lo que viene siendo un conjunto de programas que facilitan el diseño, implementación y empleo de las bases de datos, esto es más conocido como un software de base. Tiene como ventaja que puede adaptarse a necesidades especificas de un sistema.

Sus funciones principales son las siguientes:
Interpretación de comandos usados por el usuarios
Coordinación
Ventajas de los SGBD

- Independencia y seguridad de datos: Cumple con las restricciones mediante el acceso de determinados usuarios.
- Administración de los datos: Cuando los datos se encuentran centralizados, es decir compartidos, la administración de estos datos mejora, ademas de gestionar a los diferentes grupos de usuarios que utilizan la información, evitando la redundancia.
- Acceso concurrente y recuperación en caso de fallo: Los SGBD programan los accesos concurrentes a los datos de tal manera que los usuarios puedan creer que sólo tienen acceso a los datos un usuario a la vez. Además estos protegen a los usuarios de los efectos de los fallos del sistema [22].
- Reducción del tiempo de desarrollo de las aplicaciones: Los SGBD soportan funciones importantes comunes con muchas aplicaciones, mismas que tienen acceso a los datos, esto junto con la interfaz de alto nivel que facilita el rápido desarrollo de aplicaciones. Asimismo las aplicaciones de los SGBD sean más robustas.
- Acceso eficiente de los datos: Característica importante de los SGBD, ya que permite recuperar datos en dispositivos externos.
- Integridad y seguridad de los datos: Si el acceso se realiza por SGBD, se puede realizar el cumplimiento de la integridad de los datos, ademas de hacer cumplir los controles de acceso y determinar los diferentes tipos de usuarios [22].

Ejemplo de SGBD

- MySQL
- Oracle
- SQL Server 2008
- Microsotf Access

2.10.1. Lenguajes de los SGBD

Los SGBD proporcionan un servicio mediante paquetes de software los cuales permiten agilizar el almacenamiento y la búsqueda de manera muy eficiente, para esto los SGBD tienen cinco elementos principales que a continuación se muestran:

2.10.1.1. Lenguajes de los SGBD

Los lenguajes están diseñadas para cada tipo de usuario como los son administradores, diseñadores, programadores y usuarios finales. Estos lenguajes permiten que el administrador de la BD conozca como esta la estructura, sus reglas de integridad y para esto los lenguajes de clasifican en dos:

- **Lenguajes de definición de datos** (LDD o DDL por sus siglas en inglés *Data Definition Language*): Es utilizado para mostrar las vistas de los usuarios, la forma en como esta almacenada la información y en general el esquema conceptual e interno. Es empleado por los diseñadores y los administradores de las BD [19]. Estos usuarios pueden establecer permisos para cada tipo de usuario, esto dependerá de la utilización de cada uno de ellos, a esto se le conoce como Lenguaje de Definición de Vistas (VDL en Inglés *Visual Definition Language*) [1].

- **Lenguajes de manipulación de datos** (DML, en inglés *Data Manipulation Language*): Esté permite que dos de los tipos de usuarios puedan acceder a la BD, obteniendo y manipulando la información permitiendo realizar modificaciones, consultas, inserciones y eliminar datos de la BD, estos usuarios son el usuario final y los programadores que pueden realizar comandos por medio de un Lenguaje de Consultas Estructuradas o SQL [13].

- **Lenguaje de control de datos** (DCL Data Control Language): cuenta con los permisos que se le otorgar a cada uno de los usuarios para el acceso a la BD [14].

2.10.1.2. Diccionario de datos

Es el lugar donde se almacenan los datos que forman la BD , conteniendo las características lógicas de los sitios es decir, el espacio asignado por los objetos, privilegios que el usuario otorga, valores por defecto de las columnas de las tablas, definiciones como: las tablas, vistas, índices, funciones. Puede ser mencionada como un archivo de referencia generada automáticamente, de esta manera se tiene actualizada la BD [19].

2.10.1.3. Seguridad e integridad de los datos

Los SGBD proporcionan los siguientes beneficios para la integridad y seguridad de los datos.

- Protege la información ante accesos no autorizados.

- Ofrecen mecanismos de restricciones de integridad para proteger la BD, estos valores deben contener cierto tipo de restricciones de consistencia y reglas integridad, de esta manera el SGBD determina si se produjo una violación de restricción.

- Planifica y realiza copias de seguridad y restauración.

- En caso de algún daño debe recuperar la BD.
- Asegura el acceso concurrente y conserva la consistencia de los datos en caso de varios usuarios analicen la BD [19].

2.11. Java

En 1991, Sun Mycrosystems desarrollaba proyectos para dispositivos para uso del hogar como lo son los televisores, este proyecto tuvo un lenguaje llamado Oak, unos años más tarde le cambian el nombre debido al desafortunado fracaso de éste. En 1995 Java es lanzado al mundo, James Gosling padre de Java lo creó con un gran parecido al lenguaje de programación de C++, esto con el objetivo, que los programadores de C++ se sintieran cómodos al utilizar Java, una ventaja muy importante que en ese tiempo poseía Java, es que era un lenguaje simplificado y de manejo automático de memoria [31].

El éxito de Java fue gracias a Internet, poco después empezó a ser útil en los servidores superando a la competencia de ese entonces, logrando convertirse en una tecnología para aplicaciones Web, con servidores, paginas Web e incluso en la teléfonos celulares [31]. El significado de Java según la referencia [8] es café, ya que los programadores consumían grandes cantidades de café, al igual que los diseñadores.

El crecimiento de Java, ocasionó que el programa fuera ligero, facilitando su descarga desde Internet y ser portable, utilizarlo en plataformas como Windows, Linux, Unix y Macintosh. Y con todo esto, Sun en 1995 promueve a Java en la plataforma Web como para desarrollo con la versión JDK 1.0 (por sus siglas en Inglés, Java Devolepment Kit), tiempo después se crea Java Soft para la continuación del desarrollo del lenguaje [8]. Grandes empresas dedicadas a la informática creyeron en el lenguaje y hasta el día de hoy Java es un lenguaje dedicado a la programación orientada a objetos (POO).

2.11.1. Características del lenguaje Java

- Es una plataforma de desarrollo y un lenguaje de programación
- El lenguaje es sencillo, por su POO
- Es interpretado: Este trabajo lo realiza la maquina virtual con la desventaja que se hace lento, pero su ventaja es que no se necesita recompilar el programa, existiendo unicamente dos etapas la compilación y la ejecución.
- Robusto: Las declaraciones de variable son obligatorias, verifica el código en las dos etapas, es decir en la compilación y la ejecución, esto para prevenir errores. También se encarga de la sobre escritura de los punteros.
- Securizado: El motor de ejecución de Java (JRE) vigila las aplicaciones, permitiendo que la seguridad de los archivos, administra la memoria.
- Independencia de arquitectura: Se genera un bytecode (lenguaje intermedio interpretado) que independiza a la GUI (gestión de interfaz gráfica). El bytecode interpreta el código y lo hace un código nativo para que la maquina virtual lo traduzca.

- Portable: es por que su lenguaje interpretado.
- Eficaz: Esto gracias a JIT (Just In Time, por sus siglas en Inglés), que permite que optimizar el código, permitiendo alcanzar un rendimiento óptimo.
- Multihilo: permite la ejecución simultánea de más de un proceso, también denominados hilos, con el propósito de aumentar la velocidad de las aplicaciones.
- Dinámico: Permite la modificación de las clases en tiempo de ejecución [8].

2.11.2. Java swing

Java Swing es un paquete que forma parte de Java Foundation Classes (JFC), este paquete ofrece herramientas para la creación de interfaces gráficas de usuarios (GUI's Graphical User Interface), junto con sus librerias. Swing surge de AWT (Abstrac Windows Toolkit), del cual existen mejoras significativas permitiendo que java swing sea una herramienta con componentes para la interfaz, que comprende botones, cajas de texto, paneles, por mencionar algunos [16].

Capítulo 3

Desarrollo del sistema

El propósito principal de la base de datos diseñada en este trabajo, es el de facilitar la consulta de información requerida en evaluaciones de certificación o evaluación del programa Educativo de Ingeniero en Computación, del CU UAEM Zumpango. Para lograr esto, se consideraron los índices principales del programa educativo, que se pueden calcular son los siguientes:

- Número de alumnos inscritos: total, por periodo, por grupo, por UA.
- Número de alumnos regulares/irregulares: por periodo, por UA, total.
- Promedio: global, por periodo, por UA, individual.
- Número de alumnos inscritos mujeres, hombres : total, por periodo, por UA.
- Número de bajas: total, por UA, por periodo,
- Número de alumnos que egresan: total, por periodo, por cohorte

El desarrollo de una BD es una etapa que pone a prueba el ingenio y las habilidades de un diseñador de BD, el sistema que diseña constará de varios conceptos que anteriormente se mencionaron. La BD se desarrolla obteniendo los requerimientos o necesidades de los interesados en el sistema o en términos empresariales que satisfaga al cliente tu necesidad, de manera que el sistema sea funcional, pero también el diseñador ofrece mejoras y opina sobre el diseño de dicho sistemas.

En el capítulo anterior se mencionaron los conceptos principales de una BD, la arquitectura, la cual será de gran utilidad, ya que en este apartado se describirá a cada atributo con su respectivo tipo de dato y uso que se le dará.

El modelo entidad - relacional y el modelo relacional darán a la BD una idea más clara del sistema que se está desarrollando, álgebra relacional, la estructura y consultas de SQL y por último el SGBD que es donde se puede visualizar una BD como un diseño de E-R. Ahora es momento de aplicar lo que anteriormente se mencionó, desarrollando una BD para gestión de alumnos.

En este capítulo está dedicado al desarrollo y diseño de una BD, aplicando los conocimientos del capítulo 2. Para esto se retomá el planteamiento del problema que el capítulo 1.

3.1. Modelo Entidad- Relacional

El modelo relacional o diagrama relacional permite que pueda mostrar las entidades de mas importancia para el estudio y diseño de una BD. Antes de realizar el diagrama se describe las características

Modelo descriptivo:

- Un alumno tiene un número de cuenta que lo identifica dentro de la institución educativa y este número de cuenta es único e irrepetible.
- Un alumno se identifica por su número de cuenta pero también por su apellidos, nombres, CURP, teléfono particular, teléfono celular, un correo electrónico y su número de seguro social.
- Las unidades de aprendizaje (UA) se identifican por un identificador de la unidad de aprendizaje, el nombre de la UA, las horas teóricas, horas prácticas, créditos totales, antecedente o subsecuentes, núcleo al que pertenece la UA y la línea de acentuación.
- Una o más UA son parte de una linea de acentuación y otras pertenecen a un núcleo de aprendizaje.
- Las UA después de ser cursadas acreditadas o no se almacenan en un Historial para respaldar a los alumnos sobre su avance escolar.
- El Historial, almacenará los datos de alumno, la UA que curse, las bajas totales y la bajas que ha tenido por cada UA, los recuse en caso de que se tenga.
- Un grupo de calificaciones se identifica por su identificador de calificación, las evaluciones son: Primer parcial, segundo parcial, ordinario, extraordinario y examen título de suficiencia.
- Las lineas de acentuación elegida por cada alumno para su especialización, esta identificada por su identificador de línea.
- Un alumno puede tener una o varias calificaciones, estas son las obtenidas en cada parcial y la calificación del examen ordinal, estas se obtiene de las unidades de aprendizaje ofertadas en un curso en cada periodo.
- Un curso esta formado por UA que se ofertan cada periodo ordinario o intensivo.
- El periodo ordinario o intensivo lo forma grupos de alumnos que se inscriben al curso y al mismo tiempo los grupos cuentan con un turno.

En la Figura 3.1 se muestra el Modelo Entidad-Relación, donde se puede expresar los puntos anteriores.

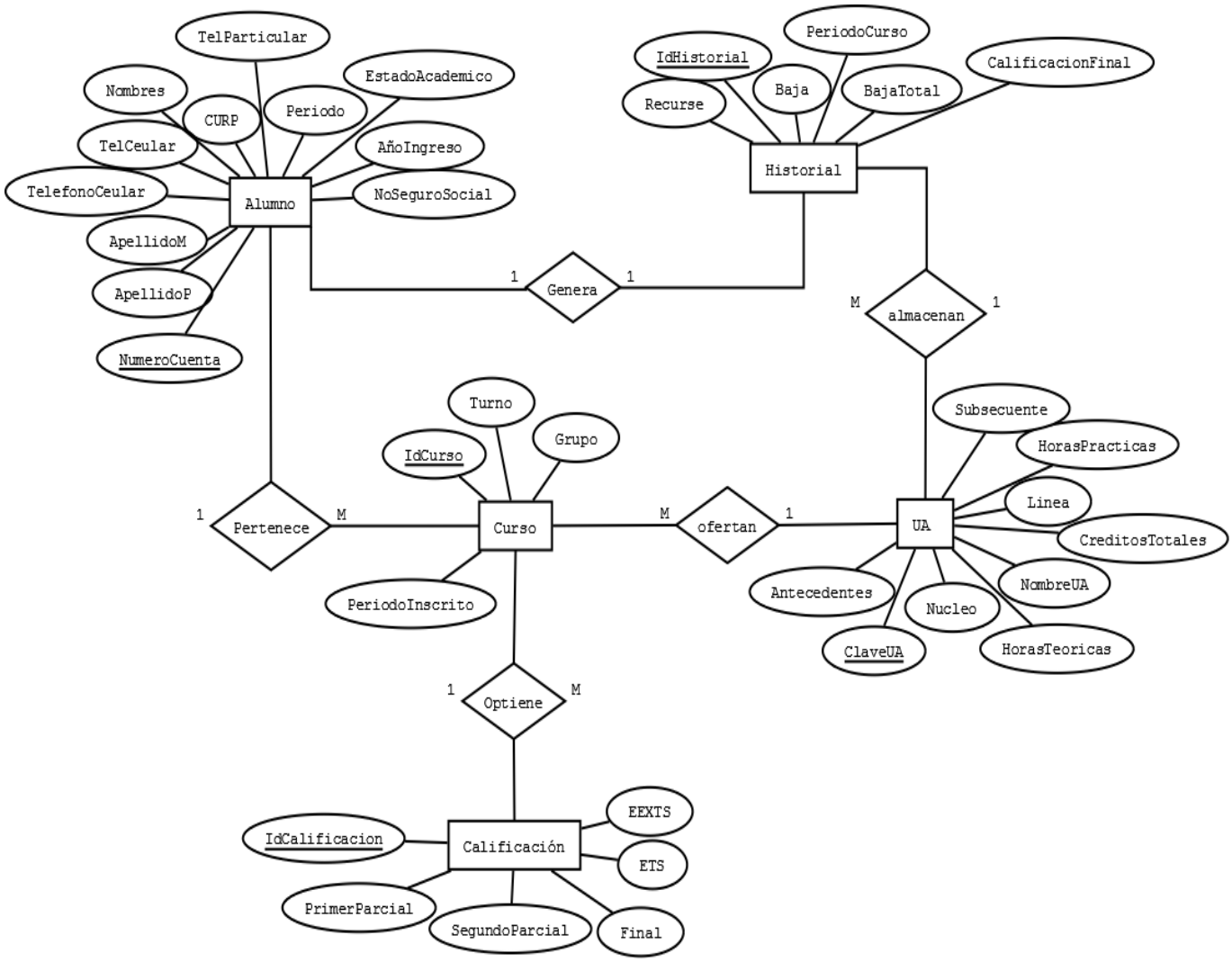


Figura 3.1: Modelo entidad-relación del sistema de desarrollo

3.2. Modelo Relacional

Después del diagrama E/R, se puede realizar el modelo relacional, para este modelo se utiliza el SGBDR(Sistemas Gestores de Base de Datos Relacionales) MySQL Workbech, que en la mayoria de los SGBD lo ocupan y se basan el en modelo relacional. Para pasar del diagrama E/R al modelo relacional es necesario conocer lo siguiente (Ver Tabla 3.1):

Modelo E/R	Modelo Relacional
Se convierte en	
Entidad	Tabla
Atributo	Columna o Campo
Identificador único	Clave primaria
Relación 1:1	La llave primaria de la tabla fuerte o de una tabla principal se ingresa a otra tabla denominada tabla débil como llave foránea
Relación 1:N	La llave primaria se inserta en la tabla de muchos como llave foránea.
Relación N:M	Se crea una nueva tabla y las llaves primarias de ambas tablas estarán como llaves foráneas en la nueva tabla.
Especial o general	La llave primaria de la tabla dominante pasa como llave foránea en las tablas dependientes.

Cuadro 3.1: Modelo E/R y Modelo relacional

En la Figura 3.2 se ilustra el modelo relacional de anera general, especificando las clases o las entidades principales que conforman el sistema.

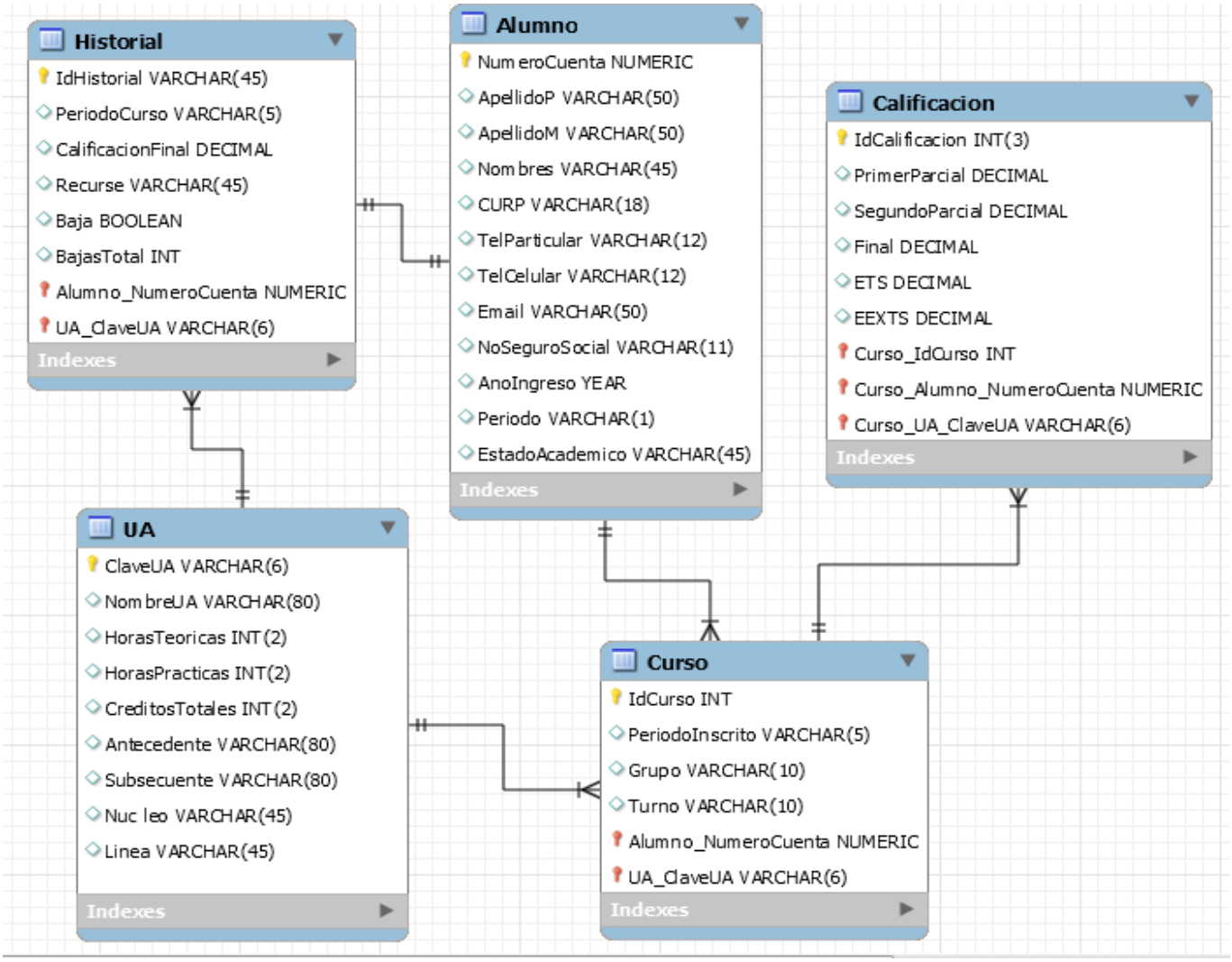


Figura 3.2: Modelo relacional del sistema de desarrollado

3.3. Descripción de tablas

Recordemos que una tabla está compuesta de filas y columnas, cada columna representa un atributo y cada fila representa la ocurrencia de una sola entidad. Es importante mencionar que el nombre de las tablas debe de ser lo más sencillo y debe cumplir con los requisitos de los identificadores que obligan la mayoría de los lenguajes de programación, es decir, no deben tener eñe (ñ) ni acentos, esto para evitar errores. Las tablas que se crearon para el desarrollo de esta BD se describen más adelante y son las siguiente: Alumno, UA, Historial, Curso y Calificacion.

En la Figura 3.3 se muestra la tabla Alumno. Dicha tabla tiene los siguientes atributos: NumeroCuenta, ApellidoPaterno, ApellidoMaterno, Nombres, CURP, TelParticular, TelCelular, Email, NoSeguroSocial, AnoIngreso, Periodo y EstadoAcademico.

Se determinó como llave o clave primaria el atributo NumeroCuenta, porque identifica a cada registro de manera única. También se pudieron considerar como claves primarias al Email y al NoSeguroSocial, ya que estos atributos también son únicos e irrepetibles, pero presenta las siguientes desventajas:

1. El atributo Email tiene una longitud extensa, y no guarda relación alguna con todos los alumnos. A diferencia del número de cuenta, que considera el año de ingreso y además lleva un orden numérico.

2. El atributo NoSeguroSocial no es considerado como un dato relevante para fines académicos, como lo es el NumeroCuenta. Además, un alumno pudiera tener un número de seguridad social asignado de manera externa a la institución.

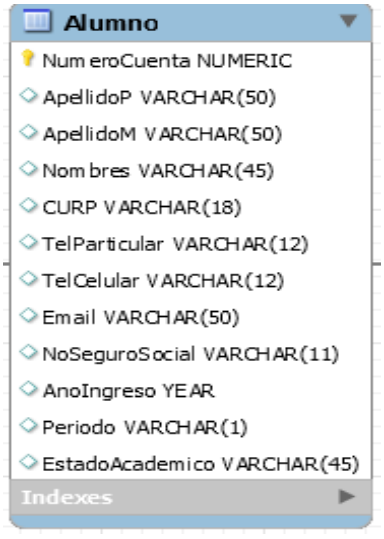


Figura 3.3: Tabla Alumno.

Un mapa curricular contiene todas las UA (unidades de aprendizaje) organizadas según el área curricular, así como también las horas teóricas, horas prácticas, total de créditos, la seriación si es que lo tuviera, la estructura por núcleo de formación. En resumen el mapa curricular organiza y ordena todo lo que se tiene que enseñar a lo largo de una preparación profesional para el aprendizaje de los alumnos.

En la Figura 3.4 se muestra la tabla UA, determinada como una tabla fuerte, que almacenará todas UA contenidas en el mapa curricular de ICO y estos sus atributos son los siguientes: ClaveUA, NombreUA, HorasTeoricas, HorasPracticas, CreditosTotales, Antecedentes, Subsecuentes, Nucleo, Linea.

Se determina como llave primaria al atributo CalveUA por las siguientes razones:

- Las UA tienen una clave única e irrepetible
- Su clave permite hacer búsquedas rápidas
- Su clave permite que su almacenamiento sea de manera organizada.

En esta tabla no cuenta con llaves foráneas, porque como se mencionó anteriormente, es una tabla fuerte, es decir, es una tabla principal, así como la tabla Alumno no cuenta con llave foránea porque también es una tabla fuerte, a partir de estas tablas se derivan las otras tablas, por lo tanto las tablas fuertes no dependen de ninguna tabla por el contrario de ellas se derivan otras tablas.

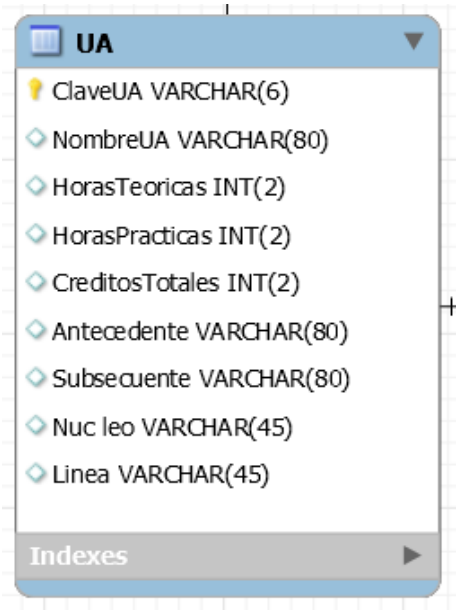


Figura 3.4: Tabla UA

Las líneas de acentuación son las opciones de especializaciones que los alumnos pueden elegir para tener un perfil específico en ICO, dentro de estas se encuentran las siguientes:

- Redes y comunicaciones
- Interacción hombre - máquina
- Desarrollo de software de aplicación
- Administración de proyectos informáticos.

Las líneas de acentuación pertenecen a la UA, algunas UA son seriadas y son necesarias para que la UA que pertenecen a la línea puedan ser cursadas, es decir, se necesitan conocimientos anteriores de otra UA, para poder cursar UA de alguna de las líneas de acentuación.

El atributo Nucleo se refiere a la forma en que como las UA están organizadas en su plan de estudios de ICO, para esto la UA está organizada en tres principales núcleos:

- Núcleo Básico: son UA elementales para proporcionara al estudiante las bases teóricas, contextuales y fisiológicas para su formación profesional.
- Núcleo Sustantivo: son conocimientos teóricos, metodológicos, técnicos e instrumentales para la comprensión de sus áreas de formación profesional.
- Núcleo Integral: deben permitir una visión de opciones ejercicios profesional y el inicio espacios laborales.

Conocer el historial académico de los alumno es tan importante, que permite conocer a que riesgos se enfrentan los alumnos y resulta de gran ayuda para evitar la deserción y el fracaso escolar. Por otro lado el historial académico es un documento que respalda la UA que el alumno ha cursado, acreditado y reprobado.

En la Figura 3.5 se muestra la tabla Historial, que almacena los datos anteriormente mencionados, esta tabla cuenta con los siguientes campos: IdHistorial, PeriodoCurso, Calificación final, Recurse, Baja, BajasTotal.

- El IdHistorial sólo se define como un identificador de la tabla pero también se define como llave primaria, porque es la ideal para referirse a la tabla.
- El campo PeriodoCurso, se define para conocer en que periodo se curso la UA cursada y acreditada, así como también conocer si se reprobó una o más UA.
- CalificacionFinal se define porque es la única calificación que se requiere para el historial.
- Recurse indicará cuales de las UA son recurse.
- Baja indica si la UA se dio de baja o no.
- BajaTotal mostrará el número de bajas de total a la UA, en caso contrario que no exista baja de UA se mostrará como que no hay ninguna UA de baja.

Las llaves foráneas pertenecen a la tabla de UA y de la tabla Alumno, esta relación existe porque para que se genere un Historial, debe existir lo siguiente:

1. El alumno debe de estar inscrito.
2. El alumno debió de cursar UA.
3. Después de cursar la UA, obtuvo calificaciones de primer parcial, segundo parcial, ordinario y en ocasiones EXT y EEXTS.

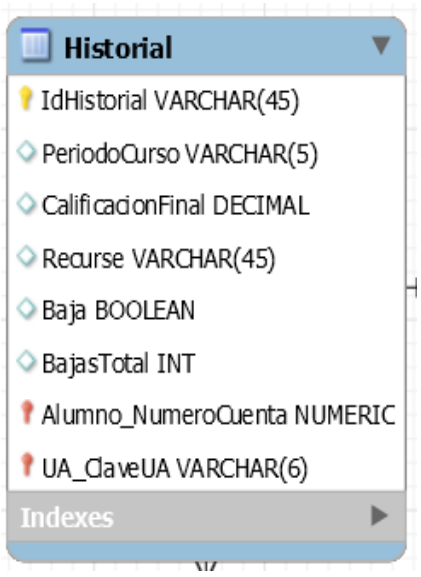


Figura 3.5: Tabla Historial

Las calificaciones son el resultado de la evaluación de cada UA, que se curso en determinado periodo o curso intensivo, las calificaciones se obtienen de 3 parciales, esto de manera normal, pero existe el caso de reprobación y se recurre a los exámenes extraordinarios, es por eso que la siguiente tabla es necesaria, se observa en la Figura 3.6, la cual contiene los siguientes atributos: IdCalificacion, PrimerParcial, SegundoParcial, Final, ETS, EEXTS.

Se determina como llave primaria al atributo IdCalificacion y como llave foránea al atributo NumeroCuenta, recordemos que este atributo en la Figura 3.3 es una llave primaria y por lo tanto en la tabla Calificacion es un llave foránea por que será la relación que tendrán estas dos tablas. Además de esa llave foránea también cuenta con la llave foránea de la tabla UA y de la tabla Curso.

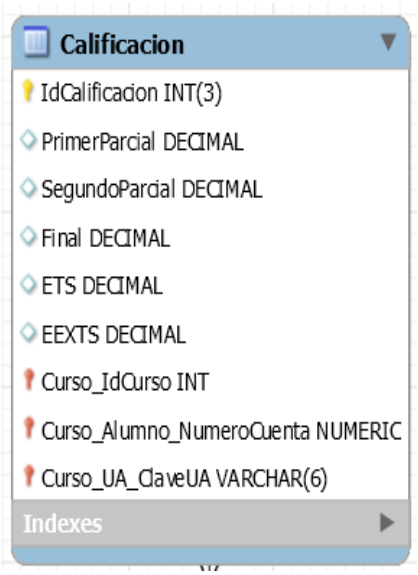


Figura 3.6: Tabla Calificación

Las UA son elaboradas de acuerdo al perfil y el plan de estudios que se requieren para el aprendizaje de los alumnos, en este caso de ingeniería en computación (ICO). Estas UA se ofertan cada periodo ordinario, intensivo y en algunos cursos especiales.

En la tabla Curso se almacenan cada UA que se oferten en cada periodo, sea ordinario o intensivo, la tabla se puede observar en la Figura 3.7, la tabla muestra los siguientes atributos: IdCurso, PeriodoInscrito, Grupo y Turno. Se determina como llave primaria al atributo IdCurso.

Como llave foránea se consideró a ClaveUA de la tabla UA, y NumeroCuenta de la tabla Alumno, un punto importante sobre las llaves foráneas es conocer que las llaves foráneas se crean a partir de las uniones, es decir, de manera gráfica como es el caso de estas tablas, las llaves foráneas no se declaran, por que el gestor de base datos crea la llave foránea mediante la relación o la unión de las tablas, caso contrario si se declaran las llaves la tabla estaría en redundancia de datos y la BD ya no estaría normalizada.

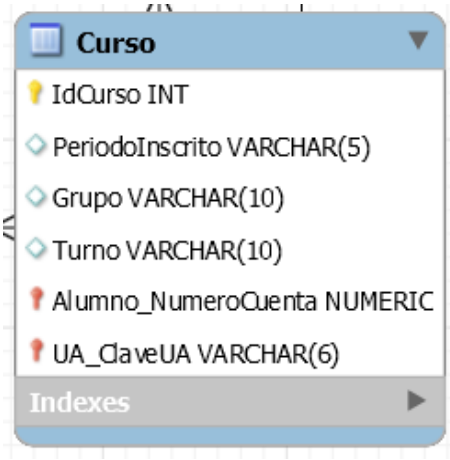


Figura 3.7: Tabla Curso

3.4. Relaciones entre tablas

En esta sección se describe como son las relaciones de las tablas mostradas en el apartado anterior, también se incluye la carnalidad o relación que existe entre las tablas, esto nos permitirá que los datos no sean redundantes.

Los requisitos importantes para que exista una relación entre tablas son:

- Un campo en común, esto es la llave primaria o llave foránea.
- El campo de registro debe ser del mismo tipo de dato.
- El tamaño del campo debe de ser del mismo tipo.

3.4.1. Relación Alumno - Historial

La primera relación que se determina son las tablas Alumno e Historial, que se muestra en la Figura 3.8.

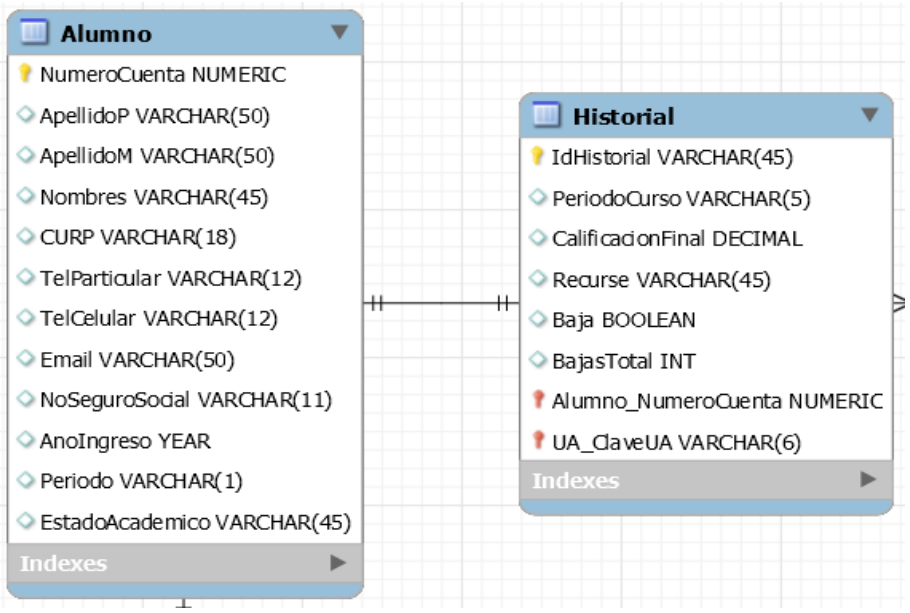


Figura 3.8: Relación Alumno - Historial

El Tabla 3.2 explica la relación entre la tablas Alumnos e Historial. La clave primaria de la tabla Alumnos, se usa como clave foránea en la tabla Historial, esto para poder identificar la calificación que le corresponde a cada alumno. Observe que en la tabla Historial se almacenan las calificaciones para cada alumno, de ahí la relación uno a uno. Es ayuda para identificar a cuál unidad de aprendizaje le corresponde una calificación obtenida por un alumno, la clave foránea NumeroCuenta se localiza en la tabla Historial.

Tablas relacionadas:	Alumnos e Historial
Tipo de relación:	1:1
Descripción de la relación:	Porque un alumno (almacenado en un registro) puede tener unicamente un historial.

Cuadro 3.2: Relación Alumno e Historial

3.4.2. Relación Alumno - Curso

La siguiente relación es Alumno y Curso (ver Figura 3.9), donde la tabla Curso se puede notar que hay dos llaves foráneas, una de la tabla UA que es la tabla con la que existe una relación y la otra de la tabla Alumno, que es quien toma el curso.

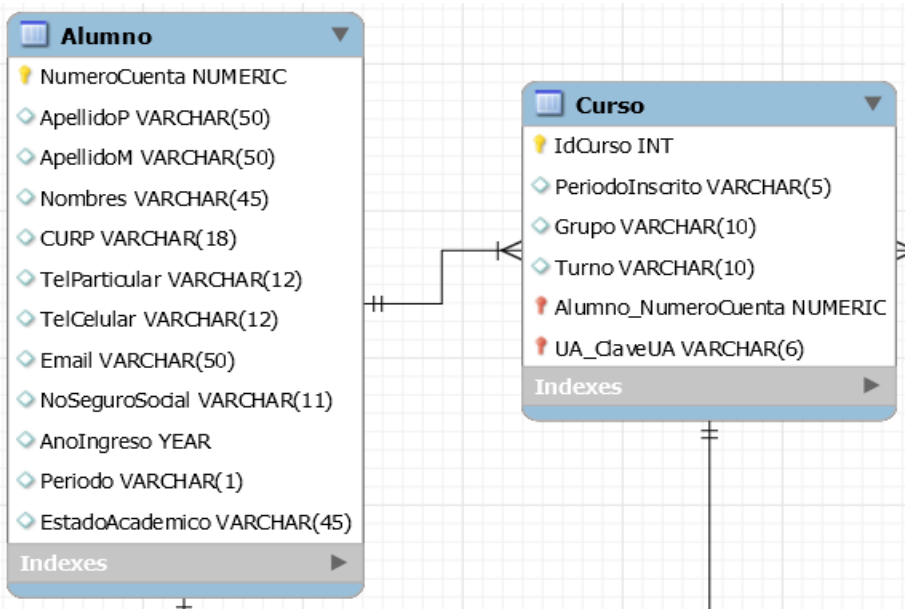


Figura 3.9: Relación Alumno y Curso

En el Tabla 3.3 indica como esta relacionado la tabla Alumno y Curso.

Tablas relacionadas:	Alumno y Curso
Tipo de relación:	1:M
Descripción de la relación:	Un alumno pertence a uno o más cursos, los cursos se ofertan cada periodo ordinario o intensivo.

Cuadro 3.3: Relación Alumno y Curso.

3.4.3. Relación UA - Curso

La siguiente relación es UA y curso, en la Figura 3.10 se puede ver como es la relación gráfica de ambas tablas. La tabla UA tiene todos los campos necesarios para almacenar UA y también se puede ver que la tabla Curso tiene llaves foráneas, esto es porque indica de que tabla provienen, recuerda que las tabla de UA es determinada como un tabla fuerte es por eso que no tiene foráneas.

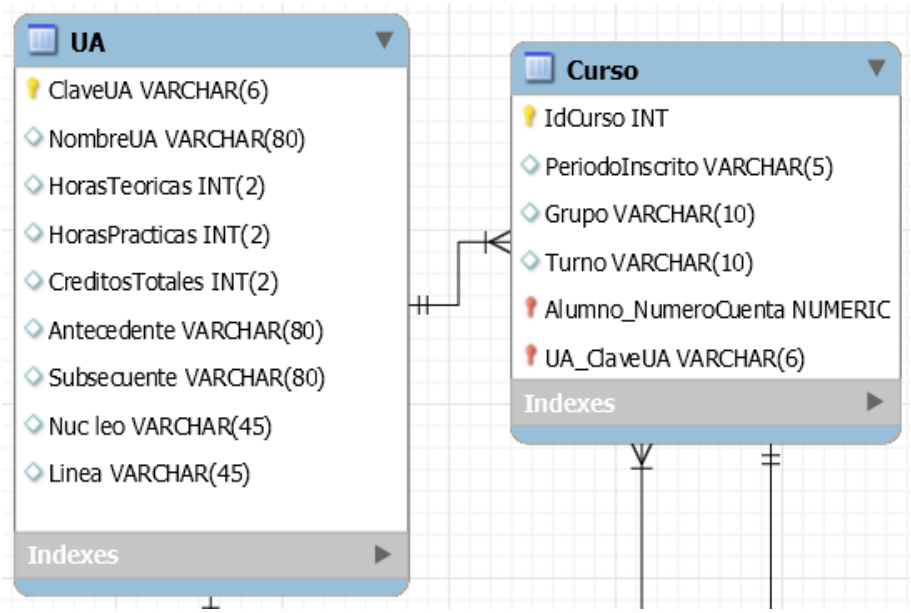


Figura 3.10: Relación UA y Curso

En el Tabla 3.4 se muestra como es la relación de UA y Curso indicando su cardinalidad y su descripción de la relación.

Tablas relacionadas:	UA y Curso
Tipo de relación:	1:M
Descripción de la relación:	Una o más Unidades de Aprendizaje ofertadas pertenecen a un Mapa curricular donde se concentran todas las UA. Las UA se ofertan: cada periodo ordinario, en periodos intensivos y en cursos especiales

Cuadro 3.4: Relación UA y Curso.

3.4.4. Relación UA - Historial

La relación de entre UA e Historial es lógica, puesto que el historial almacenará las UA que el alumno curse, es por eso que se crea esta relación. El historial también almacena la línea de acentuación que el alumno prefiera, el avance académico en forma porcentaje y los créditos que son requeridos para el término de la carrera. La tabla Historial tiene las foráneas que provienen de la tabla UA y de la tabla Alumno. En la Figura 3.11 se aprecia de manera gráfica la relación.

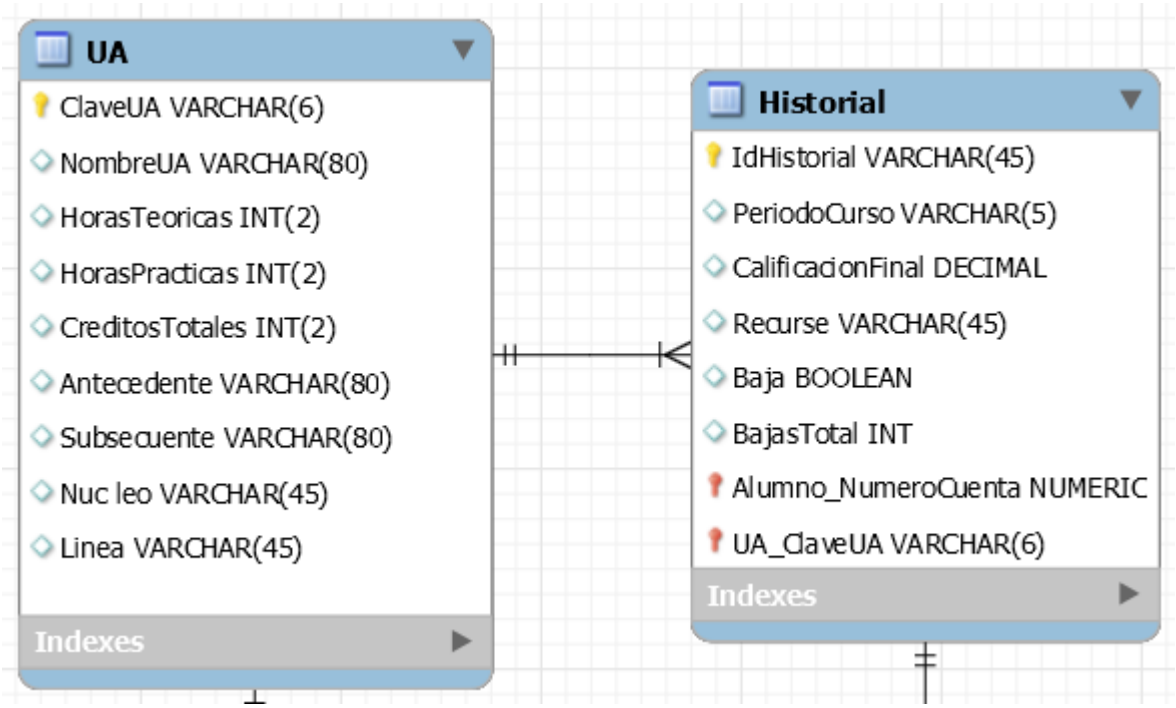


Figura 3.11: Relación UA e Historial

Para describir la relación de la Figura 3.11 es necesario ver la Tabla 3.5, donde se podrá observar cómo es la relación y que es lo que permite una relación entre estas dos tablas.

Tablas relacionadas:	UA e Historial
Tipo de relación:	1:M
Descripción de la relación:	Algunas de las UA pertenecen a las Lineas de acentuación. Estas son una especialización de la Ingeniería en computación.

Cuadro 3.5: Relación UA e Historial.

3.4.5. Relación Curso - Calificación

La relación entre la tabla Curso y Calificación es como el caso anterior, es necesario crear una relación de ese tipo ya que la UA que se encuentran en un curso y que al finalizar este, las UA deben contar con clasificaciones, en la Figura 3.12 se muestra de manera gráfica la relación.

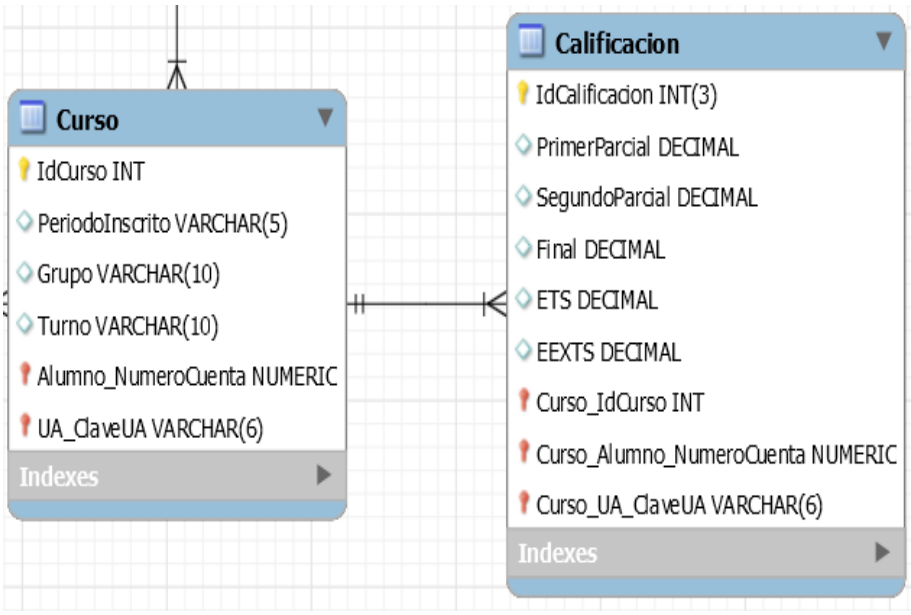


Figura 3.12: Relación Curso y Calificación

En la Tabla 3.6 se menciona como es la relación entre la tabla Curso y la tabla Calificacion, así como también la descripción entre las tablas.

Tablas relacionadas:	Curso y Calificacion
Tipo de relación:	1:M
Descripción de la relación:	Un curso tiene unidades de aprendizaje, de las cuales se obtienen calificaciones divididas en: primer parcial, segundo parcial, ordinario y extraordinario y título de suficiencia

Cuadro 3.6: Relación Curso y Calificación

3.5. Nivel lógico

Se muestra en este apartado la implementación de una BD mostrando a detalle cada atributo de las tablas y describiendo cada uno de ello. De acuerdo con el capítulo 2, donde se mencionó a detalle sobre este nivel, ahora se coloca como parte del diseño de la BD.

Para justificar el por qué de los atributos utilizados y el por qué de su longitud, se describe ne cada una de las tablas de la BD generadas.
La tabla Alumno presenta las siguientes atributos con sus siguientes descripciones (Ver Tabla 3.7 hasta la Tabla 3.18):

NumeroCuenta	
Descripción	Atributo requerido por ser único e irrepetible para cada alumno.
Uso	De uso requerido.
Tipo de dato	INT de longitud 7, porque el numero de cuenta cuenta con siete dígitos.

Cuadro 3.7: Descripción atributo NumeroCuenta

ApellidoP	
Descripción	Atributo requerido para almacenar primer apellido.
Uso	Uso requerido.
Tipo de dato	VARCHAR de longitud 50, aproximadamente el tamaño del primer apellido.

Cuadro 3.8: Descripción atributo ApellidoP

ApellidoM	
Descripción	Atributo requerido para almacenar segundo apellido.
Uso	Uso requerido.
Tipo de dato	VARCHAR de longitud 50, aproximadamente el tamaño del segundo apellidos.

Cuadro 3.9: Descripción atributo ApellidoM

Nombres	
Descripción	Atributo requerido para almacenar el nombre o los nombres de cada alumno.
Uso	De uso requerido.
Tipo de dato	VARCHAR de longitud 45 aproximadamente la longitud de más de un nombre.

Cuadro 3.10: Descripción atributo Nombres

CURP	
Descripción	Atributo requerido para almacenar Clave Única de Registro de Población.
Uso	De uso requerido.
Tipo de dato	VARCHAR de longitud 18, la CURP esta formado por 18 dígitos alfanuméricos.

Cuadro 3.11: Descripción atributo CURP

TelParticular	
Descripción	Atributo para especificar un número de teléfono particular o de casa del alumno.
Uso	De uso requerido.
Tipo de dato	VARCHAR de longitud 12 incluyendo ladas.

Cuadro 3.12: Descripción atributo TelParticular

TelCelular	
Descripción	Atributo requerido para contacto directo.
Uso	De uso requerido.
Tipo de dato	VARCHAR de longitud 12, incluye la clave lada.

Cuadro 3.13: Descripción de atributo TelCelular

Email	
Descripción	Atributo requerido para envió y recepción de correo electrónico.
Uso	De uso requerido.
Tipo de dato	VARCHAR de longitud 50, aproximada tomando en cuenta el nombre de usuario y el dominio.

Cuadro 3.14: Descripción atributo Email

NoSeguroSocial	
Descripción	Atributo requerido para conocer numero de seguro social del alumno.
Uso	De uso requerido.
Tipo de dato	VARCHAR de longitud 11, son los dígitos que componen un número de seguro social.

Cuadro 3.15: Descripción atributo NoSeguroSocial

AnoIngreso	
Descripción	Atributo requerido para conocer el año de ingreso del alumno.
Uso	De uso requerido.
Tipo de dato	YEAR de longitud 4, usando el formato AAAA

Cuadro 3.16: Descripción atributo AnoIngreso

Periodo	
Descripción	Atributo requerido para conocer el periodo en que ingreso, periodo puede ser A o B.
Uso	De uso requerido.
Tipo de dato	VARCHAR de longitud 1, unicamente se almacenara una letra.

Cuadro 3.17: Descripción atributo Periodo

EstadoAcademico	
Descripción	Atributo requerido para conocer el estado académico del alumno.
Uso	De uso requerido.
Tipo de dato	VARCHAR de longitud 45, unicamente para colocar si es regular o irregular.

Cuadro 3.18: Descripción atributo Estado acedemico

La tabla UA, a continuación se describen los atributos de esta tabla, ver Tabla 3.19 hasta la Tabla 3.26

ClaveUA	
Descripción	Atributo requerido por ser llave primaria
Uso	De uso requerido.
Tipo de dato	VARCHAR de longitud 5.

Cuadro 3.19: Descripción atributo ClaveUA

NombreUA	
Descripción	Atributo requerido para expresar el nombre completo de la UA.
Uso	De uso requerido.
Tipo de dato	VARCHAR de longitud 50.

Cuadro 3.20: Descripción atributo NombreUA

HorasTeoricas|	
Descripción	Atributo requerido para conocer cuantas horas teóricas por semana.
Uso	De uso requerido.
Tipo de dato	NUMERIC para almacenar solo un dígito

Cuadro 3.21: Descripción atributo HorasTeoricas

HorasPracticas	
Descripción	Atributo opcional en caso de que la UA requiera horas practicas .
Uso	De uso opcional.
Tipo de dato	NUMERIC almacena solo un dígito, incluso puede ser 0.

Cuadro 3.22: Descripción atributo HorasPracticas

CreditosTotales	
Descripción	Atributo opcional en caso de que la UA requiera horas practicas .
Uso	De uso opcional.
Tipo de dato	INT de longitud 2, almacena como máximo un dígito del 1 al 9.

Cuadro 3.23: Descripción atributo CreditosTotales

Antecedentes	
Descripción	Atributo opcional si es que la UA ofertada dependiera de alguna UA .
Uso	De uso opcional.
Tipo de dato	VARCHAR de longitud 100.

Cuadro 3.24: Descripción atributo

Subsecuente	
Descripción	Atributo opcional si la UA ofertada va seriada con otra UA .
Uso	De uso opcional.
Tipo de dato	VARCHAR de longitud 100.

Cuadro 3.25: Descripción atributo Subsecuente

Nucleo	
Descripción	Atributo requerido de la UA y el núcleo donde se encuentre.
Uso	De uso requerido.
Tipo de dato	VARCHAR de longitud 45 almacena lo tipos de núcleos de formación

Cuadro 3.26: Descripción atributo Nucleo

La tabla Curso es necesaria para mostrar las Unidades de Aprendizaje que se ofertan cada periodo, intensivo o curso especial, estos son los atributos de la siguiente tabla . (Ver Tabla 3.27 hasta la Tabla 3.30).

IdCurso	
Descripción	Atributo requerido por ser campo clave.
Uso	De uso requerido.
Tipo de dato	INT de longitud 2.

Cuadro 3.27: Descripción atributo IdUAOfertada

PeriodoInscrito	
Descripción	Atributo requerido para conocer el periodo al que se esta inscrito.
Uso	De uso requerido.
Tipo de dato	VARCHAR de longitud 5.

Cuadro 3.28: Descripción atributo Periodo

Grupo	
Descripción	Atributo requerido para conocer a que grupo se le están ofertando la UA .
Uso	De uso requerido.
Tipo de dato	VARCHAR de longitud 10.

Cuadro 3.29: Descripción atributo Grupo

Turno	
Descripción	Atributo requerido para conocer el turno.
Uso	De uso requerido.
Tipo de dato	VARCHAR de longitud 5.

Cuadro 3.30: Descripción atributo Turno

La tabla Historial, como se ha mencionado anteriormente, el historial almacena todo lo que el alumno acredita, reprueba, renuncia, incluyendo el porcentaje y créditos que va almacenando durante su carrera. en los siguientes Tablas se describe los atributos de esta tabla. Ver Tabla 3.31 hasta la Tabla 3.36.

IdHistorial	
Descripción	Atributo requerido por ser llave primaria.
Uso	De uso requerido.
Tipo de dato	VARCHAR de longitud 45, para almacenar el ID.

Cuadro 3.31: Descripción atributo IdHistorial

PeriodoCurso	
Descripción	Atributo requerido para conocer el periodo en el que tomo determinada UA.
Uso	De uso requerido.
Tipo de dato	VARCHAR de longitud 5.

Cuadro 3.32: Descripción atributo PeriodoCurso

CalificacionFinal	
Descripción	Atributo requerido unicamente almacenara la calificación obtenida al final.
Uso	De uso requerido.
Tipo de dato	DECIMAL, esto significa que los numero almacenados pueden ser decimales.

Cuadro 3.33: Descripción atributo CalificacionFinal

Recurse	
Descripción	Atributo requerido por ser llave primaria.
Uso	De uso requerido.
Tipo de dato	VARCHAR de longitud 45 .

Cuadro 3.34: Descripción atributo IdLinea

Baja	
Descripción	Atributo requerido para conocer si la UA se ha dado de baja.
Uso	De uso requerido.
Tipo de dato	BOOLEAN para almacenar valores de 1 o 0.

Cuadro 3.35: Descripción atributo Baja

BajaTotal	
Descripción	Atributo requerido en caso de existir la baja contabilizar cuantas bajan existen.
Uso	De uso requerido.
Tipo de dato	INT para almacenar unicamente números.

Cuadro 3.36: Descripción atributo BajaTotal

La tabla Calificación aunque no es una tabla muy grande es importante, en ella se almacenan las calificaciones de las Unidades de Aprendizaje que los alumnos cursan, ahora se mostraran los atributos y su descripción en los siguientes Tablas. Ver Tabla 3.37 hasta la Tabla 3.42.

IdCalificacion	
Descripción	Atributo requerido por ser campo clave.
Uso	Uso requerido.
Tipo de dato	INT para almacenar 3 dígitos.

Cuadro 3.37: Descripción atributo IdCalificación

PrimerParcial	
Descripción	Atributo requerido por ser primera calificación .
Uso	De uso requerido.
Tipo de dato	DECIMAL para almacenar calificación con punto decimal

Cuadro 3.38: Descripción atributo PrimerParcial

SegundoParcial	
Descripción	Atributo requerido por ser la segunda calificación.
Uso	De uso requerido.
Tipo de dato	DECIMAL para almacenar calificación con punto decimal.

Cuadro 3.39: Descripción SegundoParcial

Final	
Descripción	Atributo requerido por ser calificación decisiva.
Uso	De uso requerido.
Tipo de dato	DECIMAL para números con punto decimal

Cuadro 3.40: Descripción atributo Final

ETS	
Descripción	Atributo opcional en caso de presentar examen extraordinario.
Uso	opcional.
Tipo de dato	DECIMAL, en caso de la calificación sea con punto decimal.

Cuadro 3.41: Descripción atributo ETS

EEXTS	
Descripción	Atributo opcional en caso de presentar examen extraordinario de titulo de suficiencia.
Uso	opcional.
Tipo de dato	DECIMAL, en caso de que la calificación sea punto decimal.

Cuadro 3.42: Descripción de atributo EEXTS

3.6. Instalación MySQL Workbench

La herramienta Workbench forma parte de las aplicaciones de MySQL, que facilita el proceso de diseño, el modelado, creación de diagramas E-R y la documentación de la BD y para los administradores de BD. Workbench permite la visualización del modelado y facilita la ejecución de las consultas por medio de un editor.
Requisitos del sistema:

- Servidor MySQL
- Conexiones del cliente simultaneas
- Microsoft .NET 4.0 Framework
- Paquete redistribuible de Microsoft Visual C ++ 2013 (MSVC2013)

La descarga general de MySQL esta disponible en http://dev.mysql.com/downloads/windows/installer/, este installer puede instalar, administrar, actualizar y administrar la mayoría de los productos de MySQL Workbench, después de descargar iniciamos con la instalación, observa la Figura 3.13 donde se muestra la primera ventana con el asistente de instalción de MySQL, se da clic al botón de Next.

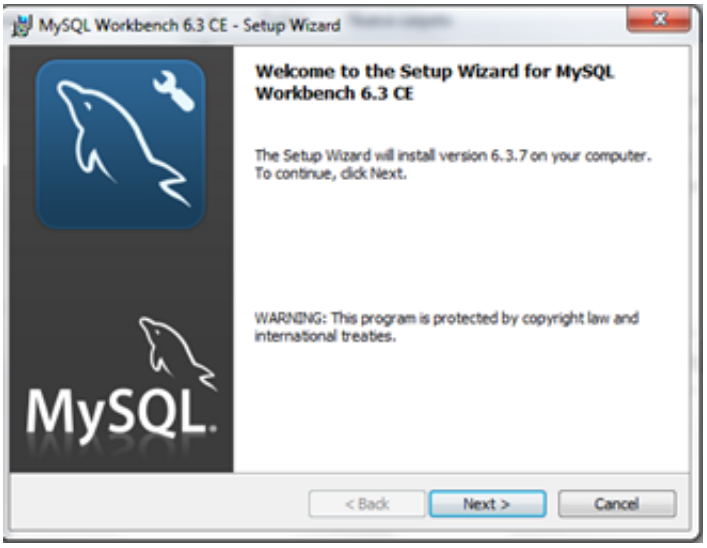


Figura 3.13: Asistente de instalación de MySQL Workbench

En la Figura 3.14 se muestra la ubicación donde se instalará MySQL Workbench

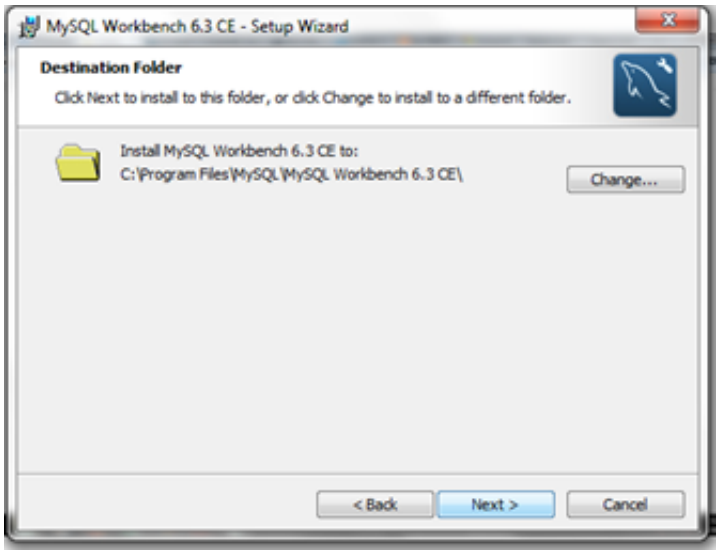


Figura 3.14: Localización de la instalación

En la Figura 3.15 se muestra el tipo de instalación, para este caso se elije la instalación completa y clic en Next.

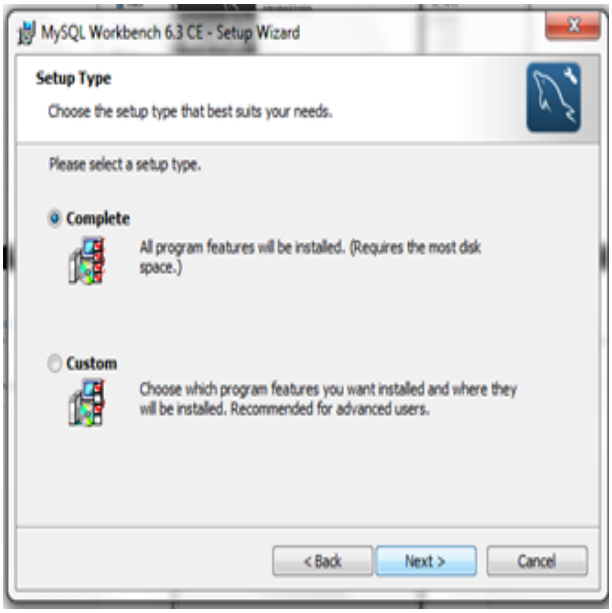


Figura 3.15: Tipo de instalación

La Figura 3.16 muestra un mensaje de confirmando que esta listo para ser instalado, clic en instalar para comenzar el proceso de instalación

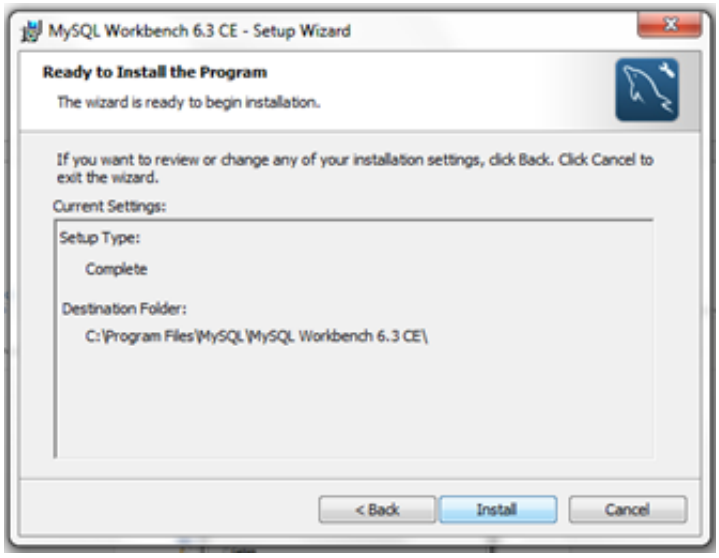


Figura 3.16: Listo para instalar

Después de terminar con la instalación el asistente muestra una ventana (ver Figura 3.17) con la completa instalación de MySQL Workbench, click en Finish.

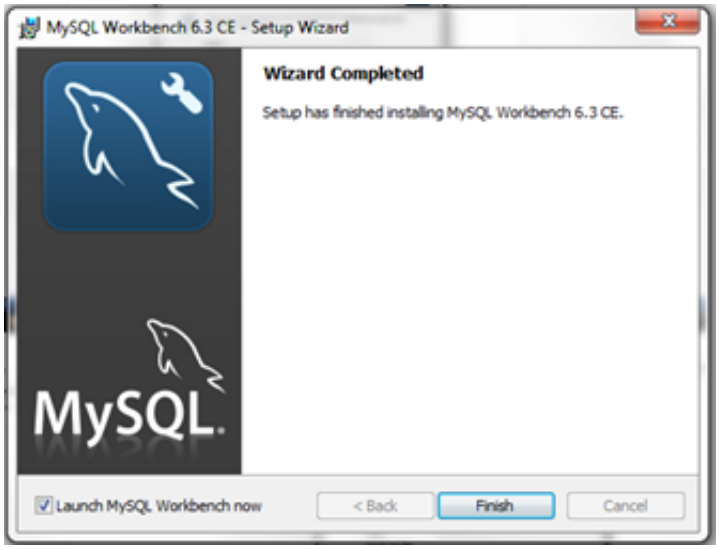


Figura 3.17: Instalación completa

3.7. Instalación y configuración del servidor MySQL

Ahora se instala el servidor MySQL, comencemos descargando el servidor de la pagina oficial de MySQL, es importante mencionar que el servidor se identifica como MySQL Community Server, se descarga para Windows y una vez que finalice la descarga se descomprime y se ejecuta como administrador y dará comienzo a la configuración como se muestra en la Figura 3.18

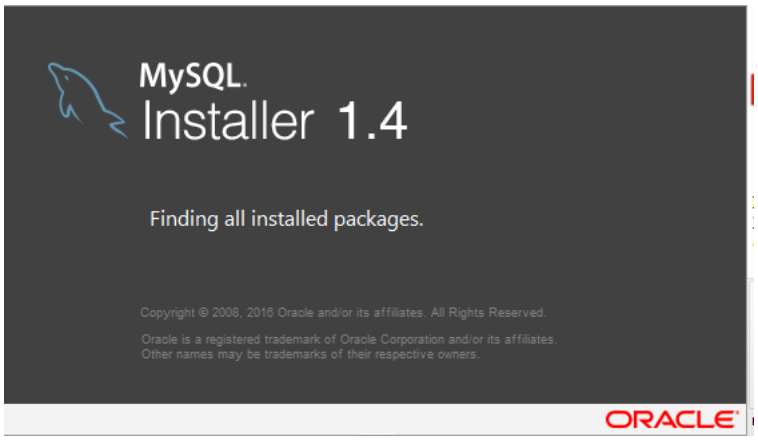


Figura 3.18: Configuración servidor MySQL

Después de encontrar todos los paquetes necesarios para la instalación se muestra la ventana que esta en la Figura 3.19, donde se muestra la versión y la arquitectura del MySQL Workbench, clic en la opción Add...

Figura 3.19: Inicio de configuración

El resultado de la Figura 3.19 se muestra en la Figura 3.20 donde se acepta los acuerdos de las licencia y clic en Next.

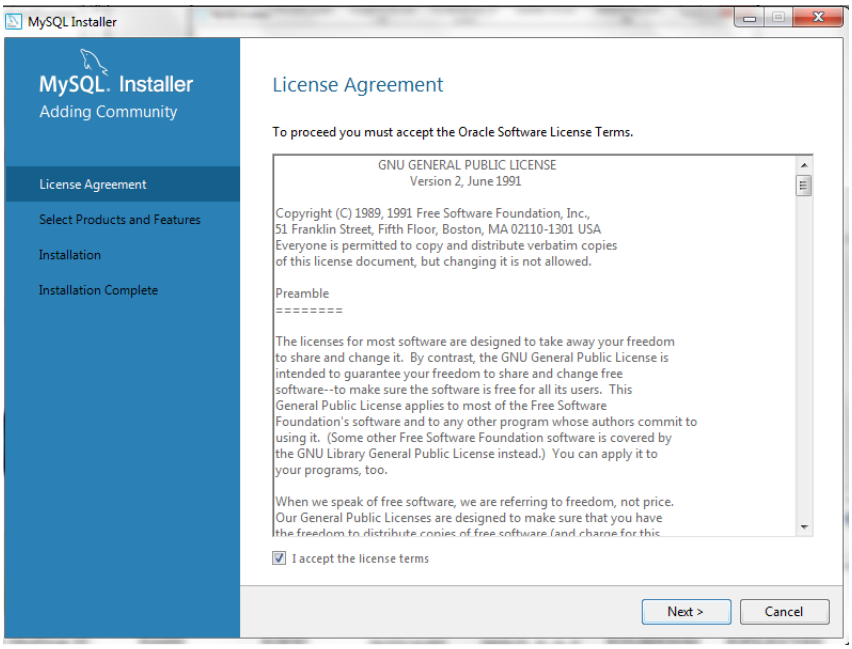


Figura 3.20: Acuerdos de licencia

Ahora se selecciona el producto (ver Figura 3.21)y las características que se desean instalar en la maquina, se despliega la primera opción que dice MySQL Servers, desplegamos dos veces más y selecionar MySQL Server 5.7.16 - X86, se agrega al siguiente panel con la flecha correcta, clic en Next

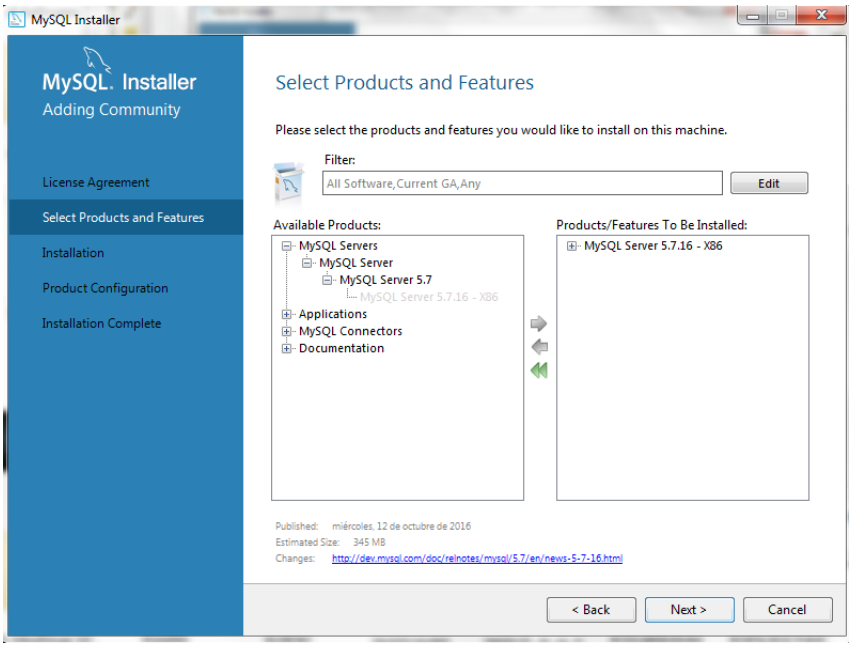


Figura 3.21: Selección de producto y características

En la Figura 3.22 muestra la ubicación donde se se guardo la instalación, esto se puede cambiar o cambiarla a una nueva, en mi caso decido dejar la misma ruta y continuar dando click en Next.

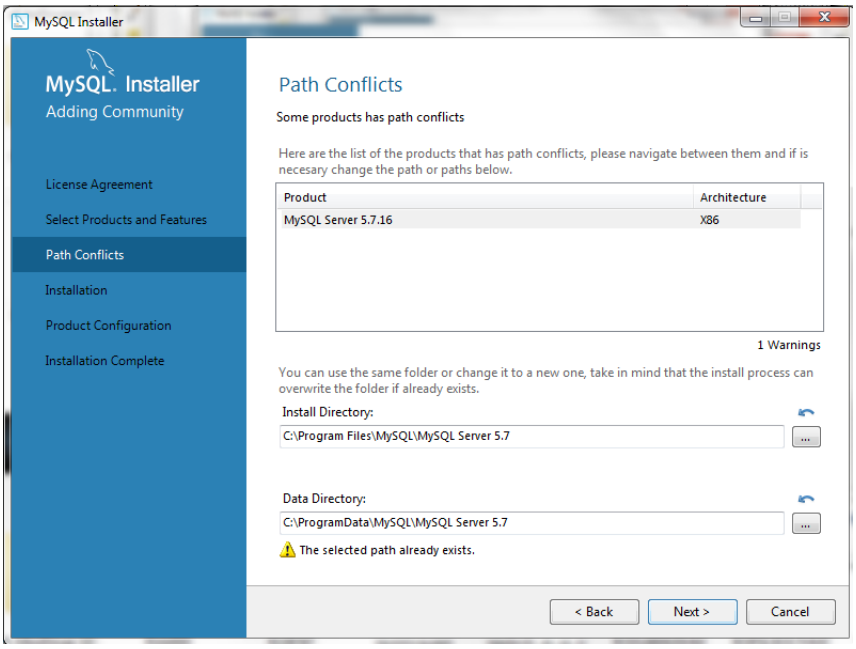


Figura 3.22: Ubicación de la instalación

En la Figura 3.23 se da comienzo al proceso de instalación dando clic en botón de Execute, esperando unos minutos para terminar la instalación y se muestra en el estado del proceso que la instalación esta completada. Después clic en Next.

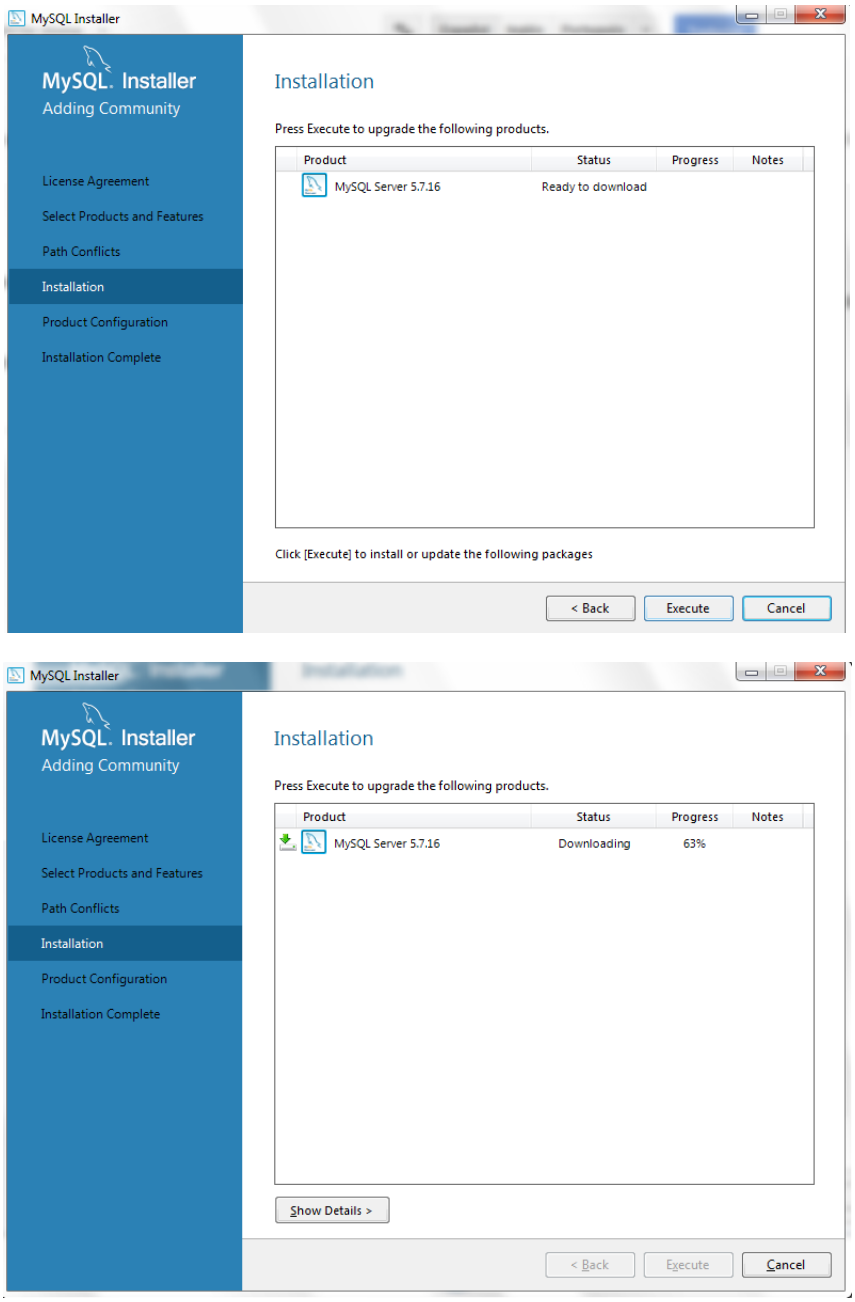


Figura 3.23: Proceso de instalación

La Figura 3.24 se muestra la configuración del producto que este caso es unicamente el Servidor de MySQL, clic en Next.

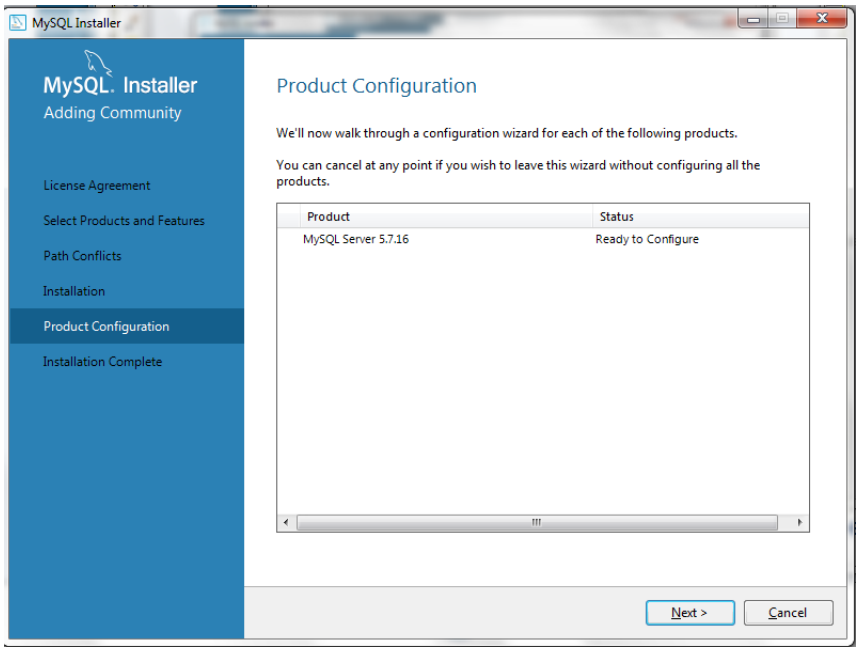
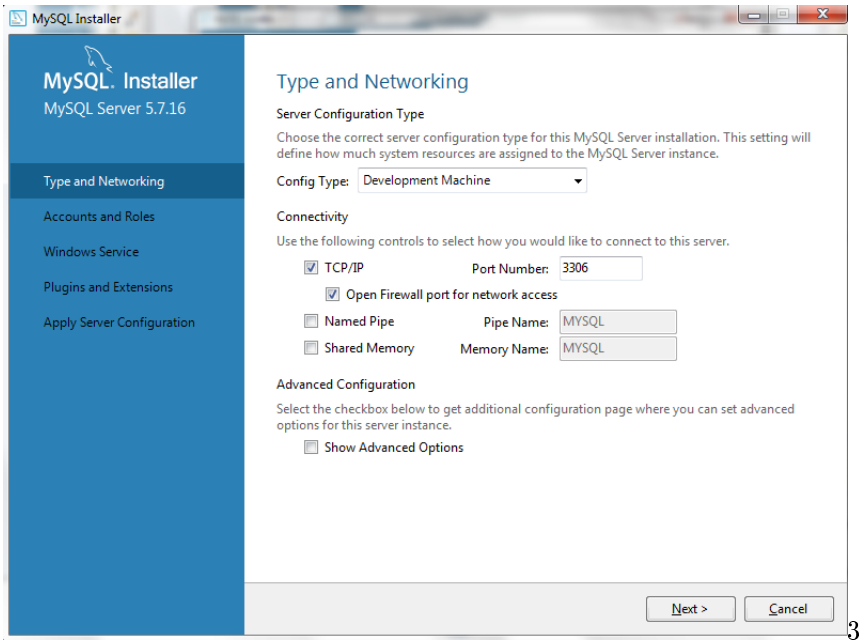


Figura 3.24: Configuración del servidor

El tipo configuración del servidor se deja tal como se muestra en la Figura 3.25, esta configuración determina la cantidad de espacio utilizado en el servidor. Dar clic en Next.



3.25

Figura 3.25: Tipo de conexión del servidor

Ahora es momento de configurar la contraseña de root, como se muestra en la Fig. 3.26 donde en la primera parte es colocar una palabra clave para poder recordar la contraseña, el siguiente apartado es el de contraseña root y la última es la confirmación de la contraseña. Después de que esté completo esto, dar clic en Next.

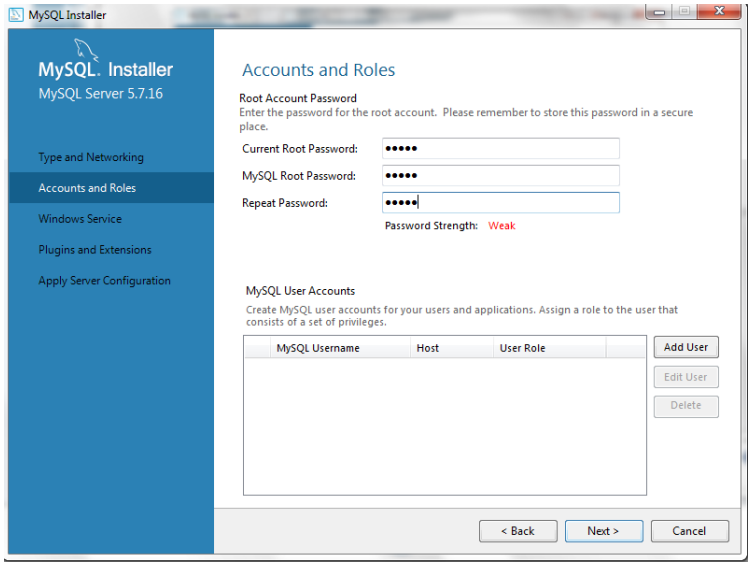


Figura 3.26: Contraseña del servidor

En los servicios de Windows se elige la configuración recomendada por defecto, dar clic en Next

para aplicar la configuración (ver Figura 3.27).

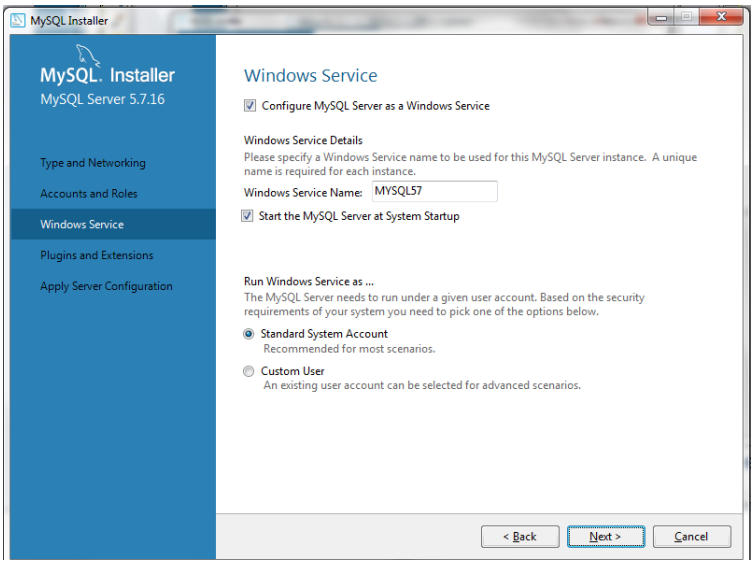


Figura 3.27: Servicio de Windows

En la Figura 3.28 se puede observar como se esta aplicando los cambios de la configuración, así como también cuando indica que la configuración esta completa, después de esto clic en Next para continuar y finalizar con la configuración.

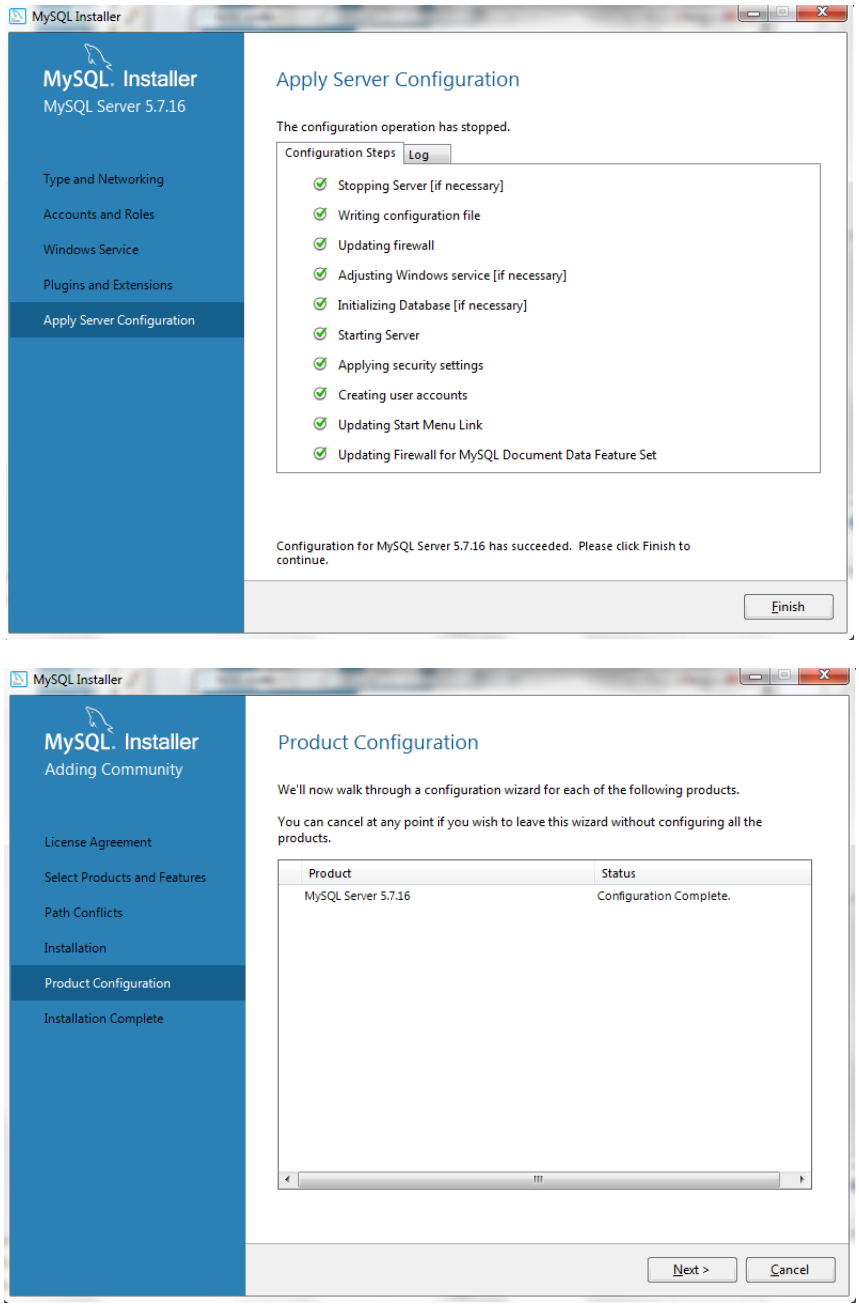


Figura 3.28: Configuración completa

3.8. Sentencias SQL

Las tablas son la base para almacenar los datos en la BD. Las tablas se dividen filas y columnas, cada fila representa uno de muchos de los datos que en ella se almacenan, cada columna representa el componente de los datos. Para entender mejor esta parte recordemos la sección anterior, que describe los atributos de cada unas las tablas y al mismo tiempo el tipo de dato que se emplea para cada dato en particular.

Cuando se crea un tabla, es necesario especificar lo siguiente:

- El nombre que se quiere asignar a la tabla
- Los nombres de los campos
- El tipo de dato que asocie de manera correcta a la información que se va almacenar, pero antes es conveniente tener una referencia de los tipos de datos que existen para BD, en Capitulo 2 se describen estos tipos de datos.
- Determinar la llave primaria, recuerda que una llave primaria siempre es única.

Para empezar y para este caso se utiliza el SGBD MySQL Workbench, que al mismo tiempo desde este gestor se manipulan los datos. Crear una BD se dice muy sencillo, pero tiene su detalle si no se domina bien la creación de tablas. Para ayudar un poco este gestor tiene la opción de crear las tablas por medio del lenguaje SQL o de manera gráfica, en este caso se crearon de manera gráfica, pero el código también se puede generar a partir del modelo relacional.

Antes de crear las tablas es necesario crear una BD especialmente para estas tablas y se llama "NewDB" y se crea con la siguiente sentencia:

```
CREATE DATABASE NewDB;
```

A partir de ahora, es posible crear las tablas, recordemos que para usar la BD creada se utiliza el comando USE, que no se puede definir como una sentencia, si no como como una opción de MySQL. Ahora continuamos con la sentencia CREATE TABLE que sirve para crear las tablas, de una manera simple se crearan las tablas con las columnas, estas columnas indicarán los atributos de cada tabla.

Comencemos con la tabla de Alumnos como se muestra en el siguiente código. Al definir cada columna se puede indicar si podrá o no tener valores nulos, la opción por defecto permite valores nulos, en caso de que no se deseé esta opción se coloca NOT NULL.

La clave primaria se define al final colocando PRIMARY KEY (' NumeroCuenta'), en este campo no puede tener valores NULL. el código se muestra en la Figura 3.29.

```
CREATE SCHEMA IF NOT EXISTS `NewDB` ;
USE `NewDB` ;
-----------------------------------------------
Table `NewDB`.`Alumno`
-----------------------------------------------
CREATE TABLE IF NOT EXISTS `NewDB`.`Alumno` (
`NumeroCuenta` DECIMAL(10,0) NOT NULL,
`ApellidoP` VARCHAR(50) NULL,
`ApellidoM` VARCHAR(50) NULL,
`Nombres` VARCHAR(45) NULL,
`CURP` VARCHAR(18) NULL,
`TelParticular` VARCHAR(12) NULL,
`TelCelular` VARCHAR(12) NULL,
`Email` VARCHAR(50) NULL,
`NoSeguroSocial` VARCHAR(11) NULL,
`AnoIngreso` YEAR NULL,
`Periodo` VARCHAR(1) NULL,
`EstadoAcademico` VARCHAR(45) NULL,
PRIMARY KEY (`NumeroCuenta`),
UNIQUE INDEX `NumeroCuenta_UNIQUE` (`NumeroCuenta` ASC))
ENGINE = InnoDB;
```

Figura 3.29: Sentencia SQL para tabla de Alumno

Después de crear la tabla de Alumnos continuamos con la de UA, esta tabla es otra de las tablas fuertes es por eso que se crean primero, el código es que sigue (ver Figura 3.30):

```
-- -----------------------------------------------------
-- Table `NewDB`.`UA`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `NewDB`.`UA` (
  `ClaveUA` VARCHAR(6) NOT NULL,
  `NombreUA` VARCHAR(80) NULL,
  `HorasTeoricas` INT(2) NULL,
  `HorasPracticas` INT(2) NULL,
  `CreditosTotales` INT(2) NULL,
  `Antecedente` VARCHAR(80) NULL,
  `Subsecuente` VARCHAR(80) NULL,
  `Nucleo` VARCHAR(45) NULL,
  `Linea` VARCHAR(45) NULL,
PRIMARY KEY (`ClaveUA`))
ENGINE = InnoDB;
```

Figura 3.30: Sentencia SQL para tabla UA

La siguiente es Curso, en esta tabla se colocan las llaves foráneas que son las relaciones entre la tabla de Alumno y la tabla de UA, estas las puedes identificar por la abreviación 'fk'. Las tablas de tipo InnDB permiten trabajar con integridad referencial, es decir, existe soporte para este tipo de claves (ver Figura 3.31). En esta tabla se puede observar que existe una columna con auto-incrementada y para que eso sea posible, el tipo de dato tiene que ser entero. Un ejemplo de esto es: Cuando se inserta una fila y es omitida su auto-incremento, el valor se calcula tomando como referencia el valor más alto y sumando una unidad.

```
-- -----------------------------------------------------
-- Table `NewDB`.`Curso`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `NewDB`.`Curso` (
  `IdCurso` INT NOT NULL AUTO_INCREMENT,
  `PeriodoInscrito` VARCHAR(5) NULL,
  `Grupo` VARCHAR(10) NULL,
  `Turno` VARCHAR(10) NULL,
  `UA_ClaveUA` VARCHAR(6) NOT NULL,
  `Alumno_NumeroCuenta` DECIMAL(10,0)NOT NULL,
PRIMARY KEY (`IdCurso`, `UA_ClaveUA`, `Alumno_NumeroCuenta`),
  INDEX `fk_Curso_UA1_idx` (`UA_ClaveUA` ASC),
  INDEX `fk_Curso_Alumno1_idx` (`Alumno_NumeroCuenta` ASC),
  UNIQUE INDEX `IdCurso_UNIQUE` (`IdCurso` ASC))
ENGINE = InnoDB;
```

Figura 3.31: Sentencia SQL para tabla Curso

A continuación se muestra el código de la tabla Calificacion, dicha tabla también tiene llaves foráneas, ver Figura 3.32.

```
-- -----------------------------------------------------
-- Table `NewDB`.`Calificacion`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `NewDB`.`Calificacion` (
  `IdCalificacion` INT(3) NOT NULL,
  `PrimerParcial` DECIMAL NULL,
  `SegundoParcial` DECIMAL NULL,
  `Final` DECIMAL NULL,
  `ETS` DECIMAL NULL,
  `EEXTS` DECIMAL NULL,
  `Curso_IdCurso` INT NOT NULL,
  `Curso_UA_ClaveUA` VARCHAR(6) NOT NULL,
  `Curso_Alumno_NumeroCuenta` DECIMAL(10,0) NOT NULL,
  PRIMARY KEY (`IdCalificacion`, `Curso_IdCurso`, `Curso_UA_ClaveUA`,
    `Curso_Alumno_NumeroCuenta`),
  UNIQUE INDEX `IdCalificacion_UNIQUE` (`IdCalificacion` ASC),
  INDEX `fk_Calificacion_Curso1_idx` (`Curso_IdCurso` ASC,
    `Curso_UA_ClaveUA` ASC, `Curso_Alumno_NumeroCuenta` ASC))
ENGINE = InnoDB;
```

Figura 3.32: Sentencia SQL para tabla Calificacion.

Y finalmente la tabla Historial que al igual que la anterior tienen claves foráneas y es una de las tablas que más datos tendrá, ver Figura 3.33.

```
-- -----------------------------------------------------
-- Table `NewDB`.`Historial`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `NewDB`.`Historial` (
  `IdHistorial` VARCHAR(45) NOT NULL,
  `PeriodoCurso` VARCHAR(5) NULL,
  `CalificacionFinal` DECIMAL NULL,
  `Recurse` VARCHAR(45) NULL,
  `Baja` TINYINT(1) NULL,
  `BajasTotal` INT NULL,
  `UA_ClaveUA` VARCHAR(6) NOT NULL,
  `Alumno_NumeroCuenta` DECIMAL(10,0) NOT NULL,
  PRIMARY KEY (`IdHistorial`, `UA_ClaveUA`, `Alumno_NumeroCuenta`),
  INDEX `fk_Historial_UA_idx` (`UA_ClaveUA` ASC),
INDEX `fk_Historial_Alumno1_idx` (`Alumno_NumeroCuenta` ASC),
  UNIQUE INDEX `Historialcol_UNIQUE` (`IdHistorial` ASC))
ENGINE = InnoDB;
```

Figura 3.33: Sentencia SQL para tabla Historial

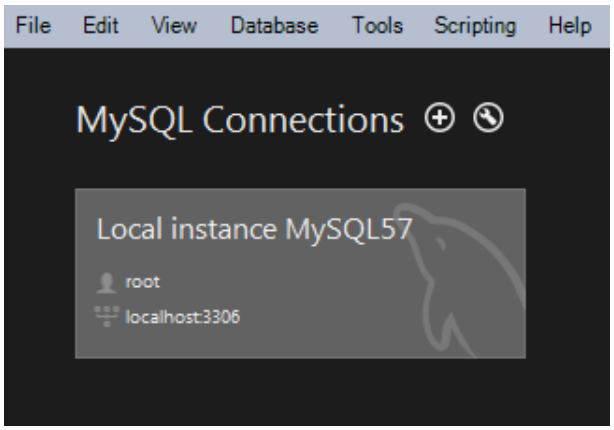


Figura 3.34: Icono de conexión

3.9. Consultas

Las consultas se utilizan para ver datos de forma diferente, estos mismos datos se pueden actualizar, modificar y analizar. Las consultas nos permiten:

- Recuperar información de una o varias tablas
- Ver registros en tablas relacionadas e incluso ver solo algunos registros
- Actualizar registros
- Calcular nuevos valores a partir de los datos, tales como promedios o frecuencias.

En este apartado se muestran las consultas que se efectuaran sobre una BD establecida, permitiendo manipular datos a partir de tablas ya creadas.

Recordando que SQL es un lenguaje de consultas y MySQL es el gestor de la BD, es decir es el software que sirve para almacenar y manipular los datos.

Teniendo establecida la conexión con el servidor, en la parte izquierda de la pantalla principal, doble clic, ver Figura 3.34.

y empezará a cargar el editor de SQL, ver Figura 3.35.

Figura 3.35: Cargardo el editor de SQL

Y ahora se muestra la ventana principal del editor de SQL (ver Figura 3.36), donde se trabaja para realizar consultas de manera gráfica, esta forma permite mejor la visualización que la linea de comandos

permitiendo tener un entorno visual que facilita el manejo de las consultas.

El editor de SQL permite crear, modificar y ejecutara consultas, agregar y eliminar datos, también se puede importar y exportar los resultados de los datos

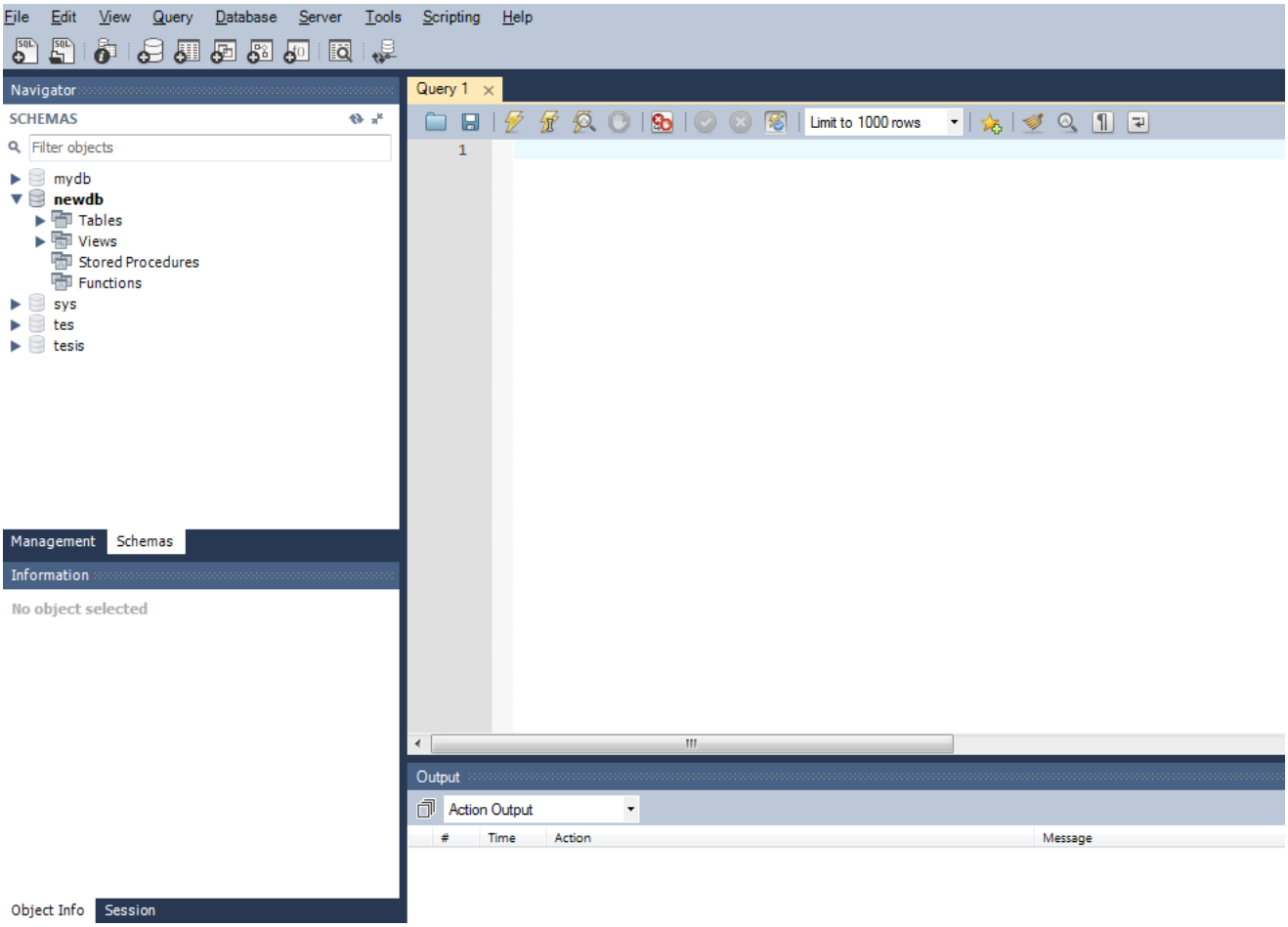


Figura 3.36: Editor SQL

El editor de SQL tiene panel y componentes, ahora se describen parte de los componentes principales del Editor de SQL, ver Figura 3.37.

1. Barra de menú
2. Navegador
3. Menú de SQL Query
4. Panel de consultas SQL
5. Salida de resultados
6. Estado de la ejecución

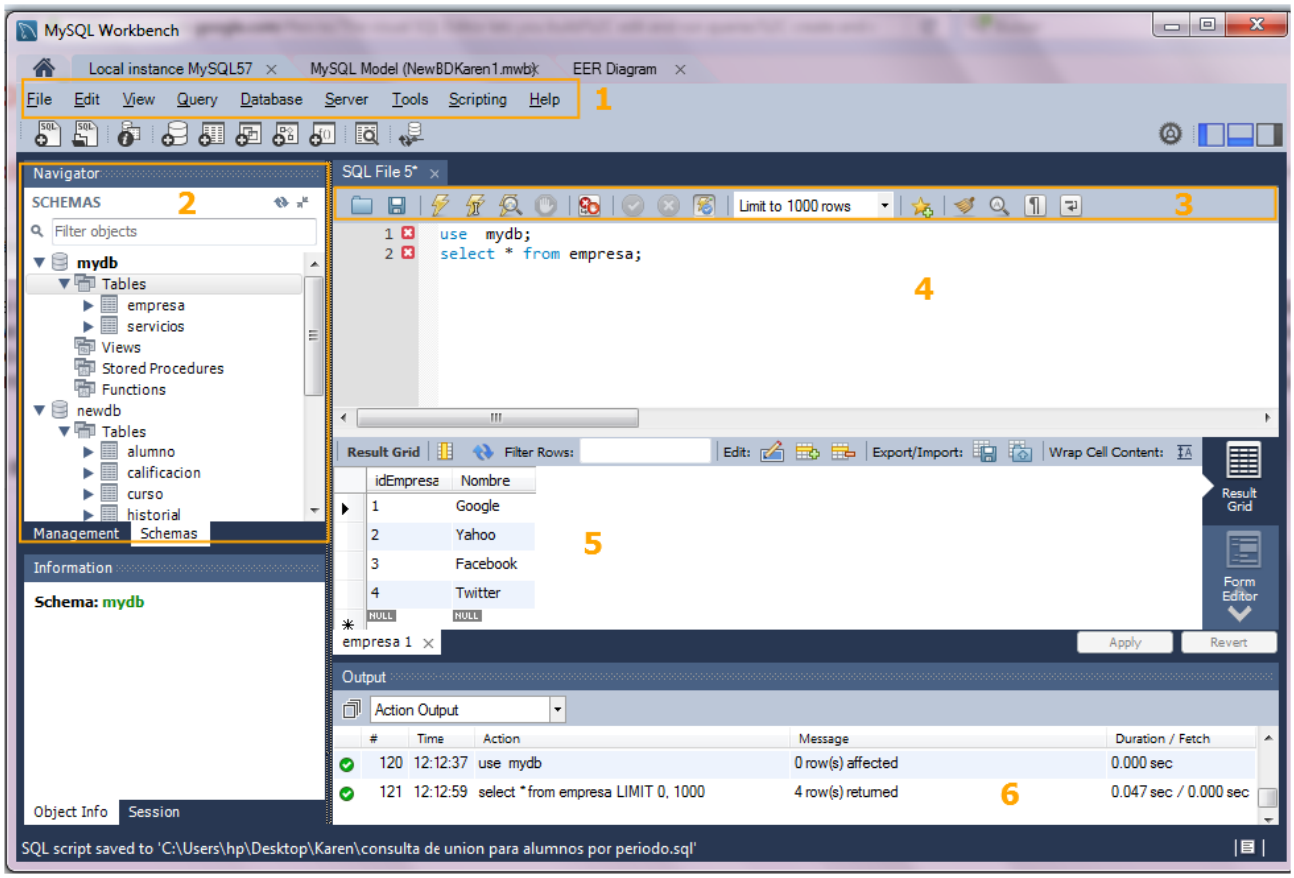


Figura 3.37: Componentes Editor SQL

La primera consulta requerida es la siguiente:

3.9.1. Buscar alumnos por número de cuenta

Esta consulta de inició es sencilla. La consulta realiza la búsqueda de alumno por numero de cuenta, recuerda que el número de cuenta es único e irrepetible para cada alumno. La estructura de la consulta es la siguiente:

1. Seleccionar en que tabla se encuentran los datos que se desea mostrar.
2. Seleccionar los campos necesarios para poder mostrar los datos, el nombre y apellidos del alumno que se desea buscar, esto se realiza por presentación formal de los datos.
3. Ahora el campo importante el numero de cuenta que se desea buscar.

El la Figura 3.38 la consulta realizada en SQL (esto se conoce en la jerga de sistemas como una consulta o Query) se muestra el Query de la consulta y en la Figura 3.39 se visualiza el resultado.

```
#selecciona alumnos por su numero de cuenta

SELECT
    Nombres, Apellidop, ApellidoM
FROM
    alumno
WHERE
    NumeroCuenta = '1134123';
```

Figura 3.38: Ejemplo de consulta búsqueda de alumnos por número de cuenta

Result Grid | Filter Rows: | Exp

Nombres	Apellidop	ApellidoM
TANIA	MONTIEL	JUAREZ

Figura 3.39: Ejemplo del resultado de la búsqueda de alumno por número de cuenta

3.9.2. Buscar alumno por Apellido Paterno

Esta consulta nos permite buscar apellidos paternos en caso de que no se conozca el nombre o el apellido materno, se puede realizar de dos maneras:

1. Buscar la cadena completa del apellido paterno completa, por ejemplo: "Hernández".
2. Buscar solo una parte del texto o cadena, por ejemplo que busque lo que contenga la cadena "Her", para esto se emplea LIKE y para completar lo que falta de la cadena se usa el comodín " %" , ver la Figura 3.40 donde se observa el uso de estos.

```
#Buscar alumno por apellido paterno

SELECT
    NumeroCuenta, Nombres, ApellidoP, ApellidoM
FROM
    alumno
WHERE
    ApellidoP LIKE '%Ro%';
```

Figura 3.40: Ejemplo de consulta buscar alumnos por Apellido Paterno

El operador LIKE busca registros o cadenas que coincidan de acuerdo a una palabra dada y el comodín " %" indica cualquier cosa que este antes o después de la cadena, En este caso, la cadena es %Ro %, busca todo lo que contenga "Ro" antes y después de cualquier cosa. El resultado se muestra en el la Figura 3.41, donde se observa que busca el apellido paterno que contenga dicha cadena.

Result Grid | Filter Rows: | Edit:

NumeroCuenta	Nombres	ApellidoP	ApellidoM
720510	LUIS	ROJAS	DESALES
1308634	CARLOS	ROJAS	GARCIA
NULL	NULL	NULL	NULL

Figura 3.41: Ejemplo del resultado de la Consulta

3.9.3. Buscar alumno por Apellido Materno

Esta consulta es parecida a la anterior unicamente se cambia el campo de búsqueda, ahora se desea buscar el Apellido Materno. Recordemos que este tipo de búsqueda existen por que se puede conocer ciertos datos de los alumnos, es por eso que se da la alternativa de realizar búsquedas con cualquier información que se tenga del alumno.

La cadena a buscar es "Gar", en la siguiente consulta que se muestra en la Figura 3.42 se muestra el query, donde mostrara todo que tenga antes y después de de la cadena.

```
#Buscar alumno por apellido materno

SELECT
    NumeroCuenta, Nombres, ApellidoP, ApellidoM
FROM
    alumno
WHERE
    ApellidoM LIKE '%Gar%';
```

Figura 3.42: Ejemplo de consulta de búsqueda de alumno por Apellido Materno

En la Figura 3.43 se observan los resultados posibles de la consulta, también se presenta el numero de cuenta, nombre, apellido paterno y el campo principal de esta consulta el apellido materno, todos estos campos no son requeridos mostrarlos, pero se muestran para darle una vista de formalidad a la búsqueda.

Result Grid | Filter Rows: | Edit:

NumeroCuenta	Nombres	ApellidoP	ApellidoM
720497	ANA	PEREZ	GARCIA
1134342	MIGUEL	RAMOS	GARZA
1192841	EDUARDO	RAMIREZ	GARCIA
1212928	LEONARDO	HERNANDEZ	GARCIA
1224786	ANGEL	MORALES	GARCIA
1308634	CARLOS	ROJAS	GARCIA
NULL	NULL	NULL	NULL

Figura 3.43: Ejemplo del resultado de la Consulta

3.9.4. Buscar alumno por Nombre

Como se ha mencionado antes el uso de LIKE, en esta consulta no es la excepción se trata se un algo similar a las anteriores consultas, de manera que es prácticamente lo mismo, aunque en cada consulta se obtienen resultados diferentes se puede observar que la estructura es la misma.

Observa la Figura 3.44 muestra la consulta y en la Figura 3.45 muestra el resultado.

```
#Buscar alumno por nombre

SELECT
    NumeroCuenta, ApellidoP, ApellidoM, Nombres
FROM
    alumno
WHERE
    Nombres LIKE '%Ana%';
```

Figura 3.44: Ejemplo de consulta búsqueda de alumno por Nombre

Result Grid | Filter Rows: | Edit:

NumeroCuenta	ApellidoP	ApellidoM	Nombres
720496	HERNANDEZ	PARRA	DIANA
720497	PEREZ	GARCIA	ANA
1224098	FLORES	VARGAS	LILIANA
1345547	SANCHEZ	CRUZ	JUANA
1407467	LOPEZ	PALMAS	ANAHI
NULL	NULL	NULL	NULL

Figura 3.45: Ejemplo del resultado

3.9.5. Buscar alumno por periodo

Si sólo se requiere buscar alumno en una BD sería muy sencillo, pero también existe la opción que se requiera saber en que periodo se encuentra los alumnos, es por eso que surge la siguiente consulta en la cual se conoce en periodo se encuentra los alumnos. Observa la Figura 3.46 donde el query de la consulta tiene la siguiente estructura:

1. Selecciona los campos que se mostrarán para dar una vista completa de los datos
2. Utilizar operador INNER JOIN para emparejar las filas de las tablas Alumno e Historial.

```
SELECT
    NumeroCuenta, Nombres, PeriodoCurso, CalificacionFinal
FROM
    alumno
        INNER JOIN
    historial ON NumeroCuenta = Alumno_NumeroCuenta;
```

Figura 3.46: Consulta de búsqueda de alumno por periodo

El resultado de la operación de INNER JOIN se muestra en la Figura 3.47.

Result Grid | Filter Rows: | Export: | Wrap Cell Co

NumeroCuenta	Nombres	PeriodoCurso	CalificacionFinal
720497	ANA	2007B	8
720510	LUIS	2011B	10
1345876	PEDRO	2012A	9
1224786	ANGEL	2012A	9
1224098	LILIANA	2012B	9
720497	ANA	2012B	9
1323074	DANIEL	2011A	9
1134342	MIGUEL	2011A	9
1224098	LILIANA	2012B	9
720496	DIANA	2011B	6
1224786	ANGEL	2011B	9
720510	LUIS	2014B	10
1134123	TANIA	2011B	9
1325548	BEATRIZ	2011B	9
1325548	BEATRIZ	2011B	9
1325548	BEATRIZ	2012A	9

Figura 3.47: Resultado de la consulta

3.9.6. Buscar alumno por UA

La siguiente consulta trata de mostrar a los alumnos que se encuentran en inscritos en una UA, ver Figura 3.48, para esto se utiliza la sentencia INNER JOIN que permite emparejar o juntar las tablas de manera mas sencilla para realizar las consultas.

```
SELECT
    PeriodoInscrito, ClaveUA, NombreUA, NumeroCuenta, Nombres
FROM
    curso
        INNER JOIN
    UAS ON UAS_ClaveUA = ClaveUA
        INNER JOIN
    alumno ON Alumno_NumeroCuenta = NumeroCuenta;
```

Figura 3.48: Ejemplo de consulta buscar alumno por UA

Como se puede observar en la Figura 3.49 se muestra el resultado de la consulta utilizando INNER JOIN, donde se observa a los alumnos que se encuentran inscritos en el periodo, también muestra el nombre de la UA, su clave, el nombre del alumno y su número de cuenta.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

PeriodoInscrito	ClaveUA	NombreUA	NumeroCuenta	Nombres
2012B	L41101	CALCULO 2	720497	ANA
2013B	L41071	I C Y C DE S O	720496	DIANA
2016B	L41061	BD AVANZADAS	720488	BRENDA

Figura 3.49: Ejemplo de resultado de consulta

3.9.7. Buscar alumno por Grupo

En ocasiones los alumno se encuentran en situaciones de repetir UA, por lo tanto se encuentran en grupos diferentes, por la necesidad de acreditar la UA que les falta, esta UA se ofertan en otros periodos. Es por eso que la siguiente consulta permite encontrar a los alumnos que se encuentran en diferentes grupos, la siguiente Figura 3.50 muestra el query que proporciona esa búsqueda de alumnos por grupo.

```
SELECT
    NumeroCuenta, nombres, grupo
FROM
    alumno
        INNER JOIN
    Curso ON NumeroCuenta = Alumno_NumeroCuenta;
```

Figura 3.50: Ejemplo de consulta de alumnos por grupo

El resultado de la consulta anterior es la que se muestra en la Figura 3.51.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

PeriodoInscrito	ClaveUA	NombreUA	NumeroCuenta	Nombres
2012B	L41101	CALCULO 2	720497	ANA
2013B	L41071	I C Y C DE S O	720496	DIANA
2016B	L41061	BD AVANZADAS	720488	BRENDA

Figura 3.51: Ejemplo de resultado de la búsqueda de alumnos por grupo

3.9.8. Buscar alumno por Turno

Así como los alumnos pueden estar en grupos diferentes, también pueden encontrarse en turno distintos, es decir que cursen UA en un turno matutino o en turno vespertino, esto por la modalidad en la que se encuentre el centro universitario.

La siguiente consulta muestra precisamente el turno en el que se encuentran los alumnos, ver Figura 3.52.

```
SELECT
    NumeroCuenta, nombres, Turno
FROM
    alumno
        INNER JOIN
    Curso ON NumeroCuenta = Alumno_NumeroCuenta;
```

Figura 3.52: Ejemplo de consulta de alumnos por Turno

En la Figura 3.53 muestra el resultado de la anterior consulta, mostrando el número de cuenta del alumno, el nombre y el turno en el que se encuentra el alumno.

Result Grid | Filter Rows:

NumeroCuenta	nombres	Turno
720497	ANA	TM
720496	DIANA	TM
720488	BRENDA	TV

Figura 3.53: Ejemplo de resultado de la consulta Alumnos por Turno

3.9.9. Buscar alumnos por año de ingreso

Ahora se creara una consulta donde en ocasiones es importante saber el año de ingreso de los alumno, para saber de que generación pertenecen o simple conocer el dato.

La consulta se muestra en el Figura 3.54, donde únicamente se puede observar que la consulta se realiza en la tabla de Alumno, y se muestran lo datos de Número de cuenta, Nombres, y Año de ingreso, recordemos que la "ñ" y los acentos no están comprendidos en el idioma Inglés, es la razón por la cual no se coloca.

```
#buscar alumnos por año de ingreso
SELECT
    NumeroCuenta, Nombres, AnoIngreso
FROM
    Alumno where AnoIngreso = '2012';
```

Figura 3.54: Ejemplo de la consulta alumnos por año de ingreso

Se observa en la Figura 3.55, el resultado de la consulta muestra a todos los alumnos que se tengan almacenados en la BD, así como también su número de cuenta y su año de ingreso.

Result Grid | Filter Rows: | Edit:

NumeroCuenta	Nombres	AnoIngreso
1212928	LEONARDO	2012
1223654	RAUL	2012
1223675	DAVID	2012
1224098	LILIANA	2012
1224786	ANGEL	2012
NULL	NULL	NULL

Figura 3.55: Ejemplo de resultado de la consulta Alumnos por año de ingreso

3.9.10. Calcular promedio de un alumno

El promedio se conoce la cantidad o cifra que resulta del aprendizaje de los alumnos en cualquier UA, es por eso que se requiere conocer el promedio de los alumnos y finalmente esta cifra da como referencia el aprovechamiento de los alumnos. En la Figura 3.56 se muestra la consulta utilizando la función AVG que nos permite calcular el promedio de cantidades mediante la especificación de los campos. En este caso sólo se requiere el promedio de un alumno.

```
SELECT
    AVG(CalificacionFinal), Alumno_NumeroCuenta
FROM
    Historial
WHERE
    Alumno_NumeroCuenta = '0720510';
```

Figura 3.56: Ejemplo de la consulta promedio de alumno

En la Figura 3.57 se muestra el resultado de la consulta anterior mostrando unicamente el promedio del alumno con numero de cuenta "0720510".

Result Grid | Filter Rows: | Exp

AVG(CalificacionFinal)	Alumno_NumeroCuenta
9	720510

Figura 3.57: Ejemplo de resultado de promedio de alumno

3.9.11. Calcular promedio por periodo

Se puede calcular el promedio de calificaciones no sólo de un alumno en particular, sino de todos los alumnos inscritos en un periodo. Si no también, el del periodo para eso nuevamente la función AVG muestra el promedio del periodo que deseamos saber, como se muestra en la Figura 3.58.

```
SELECT
    AVG(calificacionFinal), PeriodoCurso
FROM
    historial
WHERE
    PeriodoCurso = '2011B';
```

Figura 3.58: Ejemplo de consulta promedio por grupo

En la Figura 3.59 se muestra el resultado de promedio por periodo, en este caso el periodo es "2011B" , y el resultado es la suma de las calificaciones que en ese periodo se encuentren registradas en la BD.

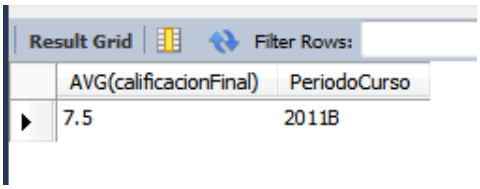


Figura 3.59: Ejemplo de resultado de promedio por periodo

3.9.12. Calcular promedio por UA

Las calificaciones que se obtienen en las UA son el resultado de la competencias que se evalúan para cuestiones académicas, la importancia de conocer el promedio de la UA es para conocer cuanto conocimiento se adquiere en cada ocasión que estas UA se ofertan. Es por eso que en la Figura 3.60 se observa la consulta promedio de la UA Inglés C1 con clave "L41504".

```
SELECT
    AVG(calificacionFinal), UAS_ClaveUA
FROM
    historial
WHERE
    UAS_ClaveUA = 'L41504';
```

Figura 3.60: Ejemplo de consulta promedio por UA

Para visualizar el resultado de la consulta en la Figura 3.61 se muestra la clave de la UA y su promedio.

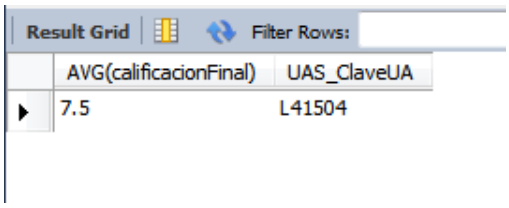


Figura 3.61: Ejemplo de resultado de promedio por UA

3.9.13. Estado académico de los alumnos

El resultado de sus calificaciones de los alumnos no son la única forma de conocer las situación real de los alumnos, existe otra forma como lo es el estado académico de los alumnos, es decir, si el alumno es regular o irregular. Los alumnos regulares son alumnos cuya situación académica no tienen UA reprobadas o tienen UA en recurse y los alumno irregulares son alumnos que son lo contrario de los regulares, es decir, les falta acreditar UA, tienen UA reprobabadas. En la Figura 3.62 se muestra la consulta.

```
SELECT
    NumeroCuenta, EstadoAcademico, PeriodoInscrito, grupo, turno
FROM
    alumno
        INNER JOIN
    curso ON NumeroCuenta = Alumno_NumeroCuenta;
```

Figura 3.62: Ejemplo de consulta estado académico de alumnos

En la Figura 3.63 se muestra el resultado de la consulta anterior, en la cual lista a todos los alumnos que estén registrados en la BD, mostrando su estado académico el periodo en el que están inscritos, grupo y turno.

Result Grid | Filter Rows: | Export: | Wrap C

NumeroCuenta	EstadoAcademico	PeriodoInscrito	grupo	turno
720497	Irregular	2012B	1A	TM
720496	Irregular	2013B	3A	TM
720488	Regular	2016B	8A	TV

Figura 3.63: Ejemplo de resultado de estado académicos

3.9.14. Alumnos inscritos por periodo

En cada periodo la inserción y deserción de alumnos es importante por que son indicadores que muestran a los académicos cuanta matricula tiene a su cargo cada periodo. La siguiente consulta muestra en forma de lista a los alumnos que se inscribieron y en que periodo (ver Figura 3.64).

```
# 15 listar alumnos inscritos por periodo
SELECT
    periodoinscrito, NumeroCuenta, nombres
FROM
    curso
        INNER JOIN
    alumno ON Alumno_NumeroCuenta = NumeroCuenta;
```

Figura 3.64: Ejemplo de consulta alumnos inscritos por periodo

En esta consulta se muestra el periodo en el que estan inscritos los alumnos mostrando también el numero de cuenta y nombre del alumno, para esto ver la Figura 3.65 donde se muestra más a detalle sobre el resultado de la consulta.

Result Grid | Filter Rows:

periodoinscrito	NumeroCuenta	nombres
2012B	720497	ANA
2013B	720496	DIANA
2016B	720488	BRENDA

Figura 3.65: Ejemplo de resultado de Alumnos inscritos por periodo

3.9.15. Alumnos inscritos por UA

Las unidades de aprendizaje que se oferta cada periodo tienen un número determinado de alumno, es decir, existe un número determinado de lugares, es por eso que las UA se ofertan cada determinado periodo, en la Figura 3.66 se observa el query de la consulta.

```
SELECT
    UAS_ClaveUA, NumeroCuenta, nombres
FROM
    curso
        INNER JOIN
    alumno ON Alumno_NumeroCuenta = NumeroCuenta;
```

Figura 3.66: Ejemplo de consulta alumnos inscritos por UA

En la Figura3.67 se muestra el resultado de la consulta, donde se muestra el nombre de los alumnos, así como su número de cuenta

Result Grid | Filter Rows:

UAS_ClaveUA	NumeroCuenta	nombres
L41101	720497	KAREN
L41071	720496	DIANA
L41504	1324086	JOEL
L41071	1425061	ANA
L41071	1223829	IVAN
L41061	720488	BRENDA

Figura 3.67: Ejemplo de resultado alumnos inscritos por UA

3.9.16. Alumnos inscritos por Línea de acentuación

El plan de estudios de la carrera universitaria requiere que el alumno elija una línea de acentuación o de especialización o comúnmente llamada línea de acentuación, la cual como su nombre lo dice es una especialización que el alumno toma en su formación profesional, la siguiente consulta (ver Figura 3.68) muestra cuántos alumnos tiene inscritos en la línea de acentuación de redes y comunicaciones (RyC)

```
Select count(*) from UAS where Linea = 'RyC';
```

Figura 3.68: Ejemplo de alumnos inscritos en linea de acentuación

Para esta consulta se emplea la función COUNT que permitirá contar los registros que tengan la línea de acentuación RyC, el resultado de dicha consulta está expresado en la Figura 3.69.

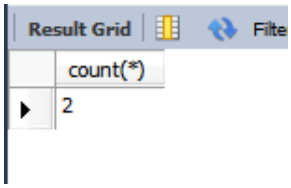


Figura 3.69: Ejemplo de resultado de Alumnos inscritos por línea de acentuación

3.9.17. Calificaciones por Número de Cuenta

Una de las maneras de conocer las calificaciones es por su número de cuenta, esto para evitar llenar de datos no tan necesarios. la siguiente consulta muestra las calificaciones por parciales de una UA, de un alumno en especifico, dichas calificaciones obtenidas durante un periodo. Ver Figura 3.70 que muestra el código de la consulta.

```
SELECT
    NumeroCuenta,
    nombres,
    Curso_UAS_ClaveUA,
    PrimerParcial,
    SegundoParcial,
    Final
FROM
    alumno
        INNER JOIN
    calificacion ON NumeroCuenta = Curso_Alumno_NumeroCuenta
WHERE
    NumeroCuenta = '1324086';
```

Figura 3.70: Ejemplo de consulta de calificaciones por número de cuenta

El resultado de la consulta se muestra en la Figura 3.71.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

NumeroCuenta	nombres	Curso_UAS_ClaveUA	PrimerParcial	SegundoParcial	Final
1324086	JOEL	L41504	8	7	7

Figura 3.71: Ejemplo de consulta de resultado de calificaciones por número de cuenta

Capítulo 4

Pruebas y resultados realizadas al sistema

En capítulos anteriores se han realizados los pasos para realizar un sistema utilizando las BD, definiendo y estructurando un proyecto para fines de seguimiento académico de alumnos. También se ha presentado el desarrollo del proyecto de manera escrita e ilustrada para su mejor entendimiento. El análisis del proyecto fue una de las etapas principales para poder resolver el problema utilizando un análisis de ingeniería y dominio de las BD, permitiendo el desarrolló e implementación de este proyecto de tesis. También se han utilizando herramientas de software como el gestor de BD MySQL Workbench en donde se desarrollo las tablas, la inserción de los datos y el modelo relacional. Otra herramienta de software es Dia, un programa para la creación de diagramas, entre ellos los diagramas de E-R [17], NetBeans IDE 8.2 para creación de la interfaz.

En este capítulo se describe y muestra el análisis para el planteamiento del problema que se describe en el capítulo 1. También se mostraran las pruebas y resultados que permiten la conclusión del capítulo 3, en donde sólo se muestra el desarrollo de la BD.

4.1. Pruebas

Recordemos que este proyecto se realizó por la necesidad que tiene la coordinación de Ingeniería en Computación del CUZ para el seguimiento académico de alumnos, para esto, en el capítulo 1 se encuentra el planteamiento del problema, el cual se retoma en este apartado para demostrar los resultados del proyecto.

El primer punto de referencia es terminar con el almacenamiento y control en Excel, a partir de este punto comienza el análisis para diseñar la BD.

El análisis queda de las siguiente manera:

- Analizando los requerimientos del sistemas y las necesidades que hay que cubrir se puede definir que es necesario almacenar a los alumnos de Ingeniería en computación.
- Conocer a detalle el mapa curricular que comprenden las UAS para la formación de los futuros

ingenieros.

- Conocer y analizar como se almacena las calificaciones de los alumnos.
- Analizar los periodos en los que se ofertan las UAS.
- Para que los alumnos tengan un respaldo de su trayectoria académica es necesario establecer que los alumnos lleven un historial académico, el cual se almacena actualmente en archivos con extensión .txt.
- Las UAS que se cursan cuentan con un número determinado de horas teóricas, horas prácticas y numero de créditos totales, los cuales son necesarios para indicar que porcentaje de la trayectoria se lleva cursada.

Después del análisis continuamos a crear la BD en MySQL Workbench, esta BD estará vacía, por que aún no contiene tablas, Workbench permitirá la manipulación de la BD para crear las tablas. La manera de comprobar que la BD esta bien, es que cumpla con los requisitos necesarios para solucionar un problema, que para este caso es en control de los alumnos de PE de ICO.

Para insertar los datos en las tablas es importante mencionar que existen tablas fuertes y tablas débiles de manera que, la inserción de los datos sera a partir de las tablas fuertes, posteriormente las tablas débiles, recordemos que las débiles depende las fuertes. esta es la manera de probar la BD en cuestión de inserción de los datos.

Teniendo los datos insertados, podemos empezar con las consultas en la BD, estas son algunas de muchas que se pueden obtener pero esto dependerá del diseño y las características para las cuales esta diseñada esta BD, Las consultas son enfocadas principalmente en los alumnos, en conocer datos de los alumnos, como su datos personales, calificaciones, situación académica, etc.

Se comprueba que están correcta por el propio MySQL, que se encarga de comprobar que las consultas estén correctas.

El siguiente paso es el diseño de la interfaz del sistema, la cual pretende ofrecerle al usuario una manipulación cómoda y fácil, que permita que el usuario pueda manipularlo de forma rápida, se vuelve a recordar que esta interfaz uno de muchos posibles diseños que se le pueden dar al sistema para el control de alumno. El desarrollo de la interfaz esta sobre el programa de desarrollo NetBeans, que su lenguaje de programación es Java.

4.2. Resultados

Se creó una interfaz de usuario en Java para el sistema de información desarrollado. Esta interfaz cuenta con 6 pestañas, las cuales contienen varias consultas agrupadas en categorías de consultas relacionadas. Las de estas Figuras 4.1 y 4.2 muestran un ejemplo de los contenidos de dos pestañas.

.

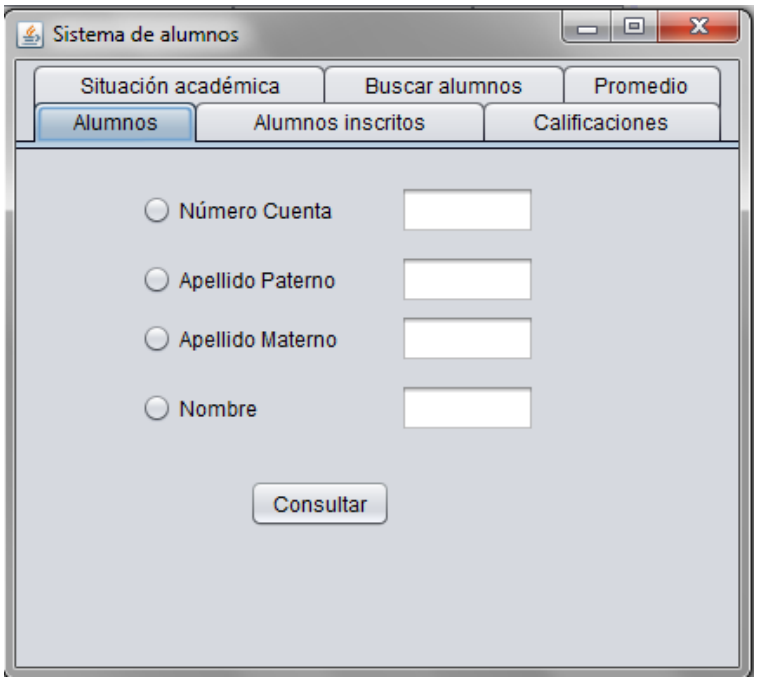


Figura 4.1: Interfaz de sistema para control de alumnos

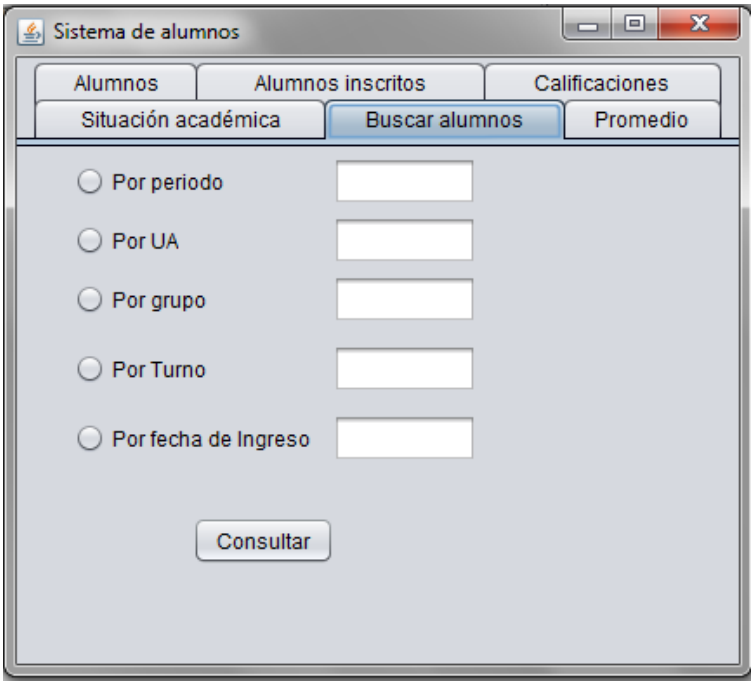


Figura 4.2: Interfaz del sistema de control de alumnos.

Para probar el sistema, se agregaron 20 alumnos de prueba, 12 UA, y 4 líneas de acentuación. Con

el fin de comprobar el funcionamiento del sistema, , se realizaron todas las consultas que se solicitaron por parte del Coordinador actual de Ingeniería en Computación del CUZ. De todas ellas, por facilidad de explicación, se presenta la consulta de buscar a alumnos por apellidos paterno. Se observa en la Figura 4.3 que se debe de eligir la pestaña de Alumnos, se elige la opción de Apellido paterno y se escribe en el espacio en blanco un apellido.

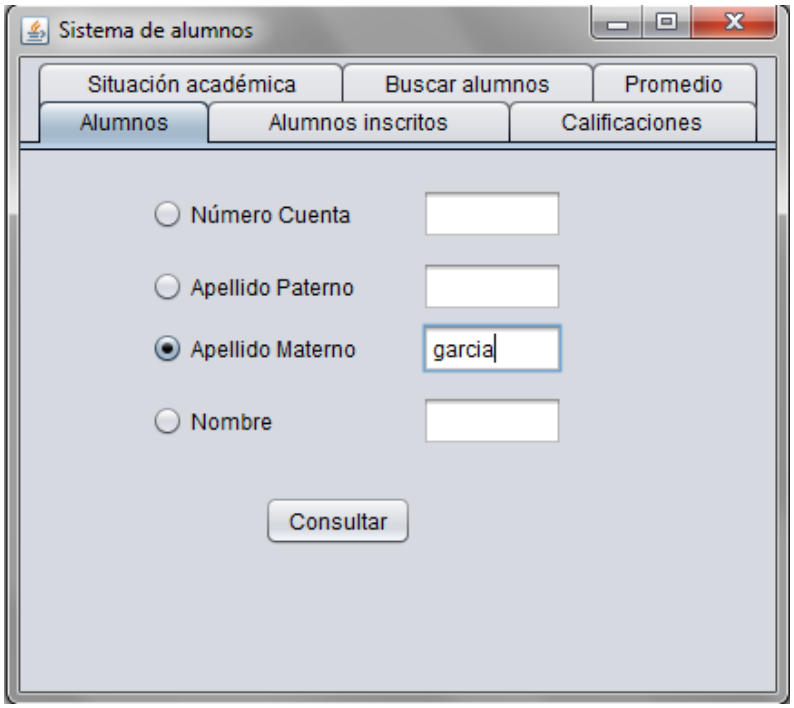


Figura 4.3: Consulta en la interfaz

En la Figura 4.4 se muestra el resultado de la consulta. En la cual se muestra todos los alumnos que tengan el apellido materno «García», incluyendo el número de cuenta el apellido paterno y el nombre.

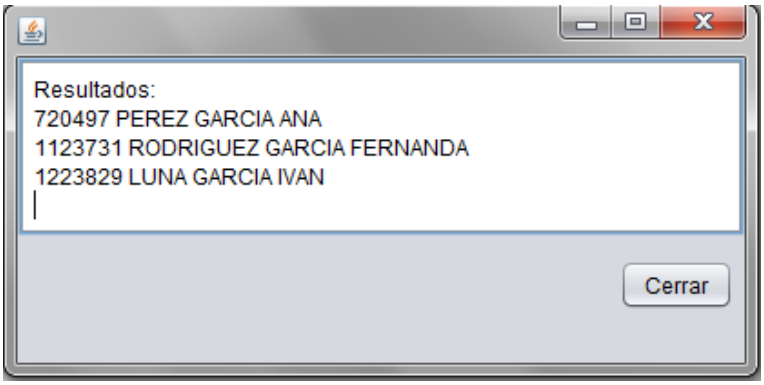


Figura 4.4: Resultado de la búsqueda.

En caso de que el usuario cometa algunas equivaciones de omisión, el sistema esta preparado para advertir al usuario que faltan datos.De manera, el sistema informa sobre algún problema que exista en caso de la consulta no se realice. Como parte de las pruebas, se cometió deliberadamente un error de omisión en cada uno de los campos. Como ejemplos, se presentan en las Figuras 4.5 y 4.6 las ventanas de información que indican que faltan datos por introducir para completar la consulta.

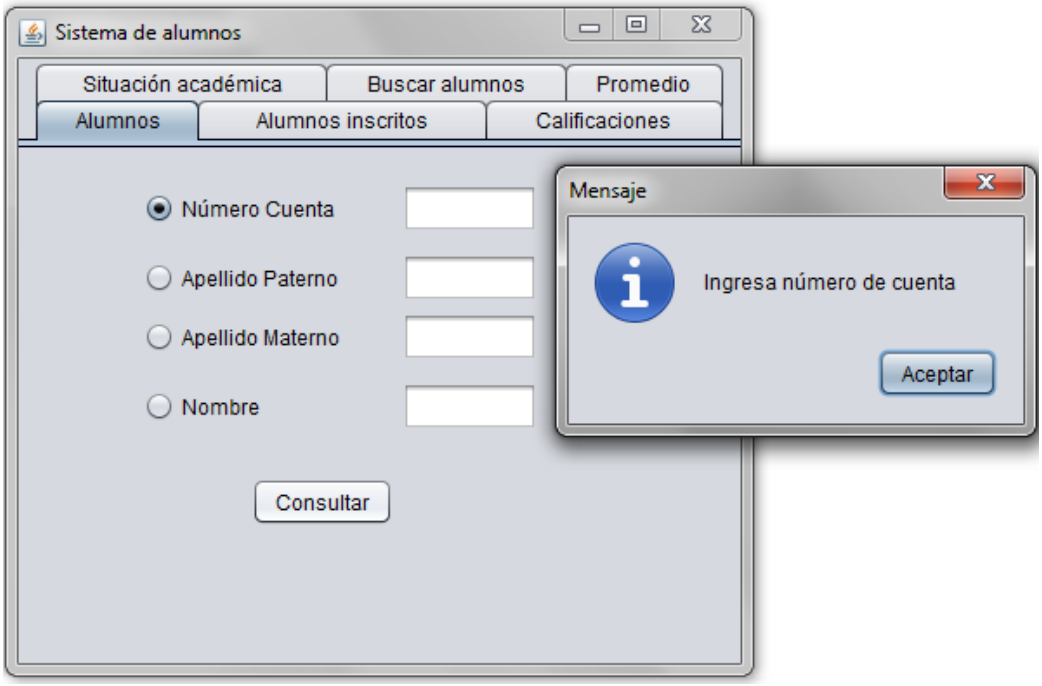


Figura 4.5: Ventana de información: Ingresar número de cuenta

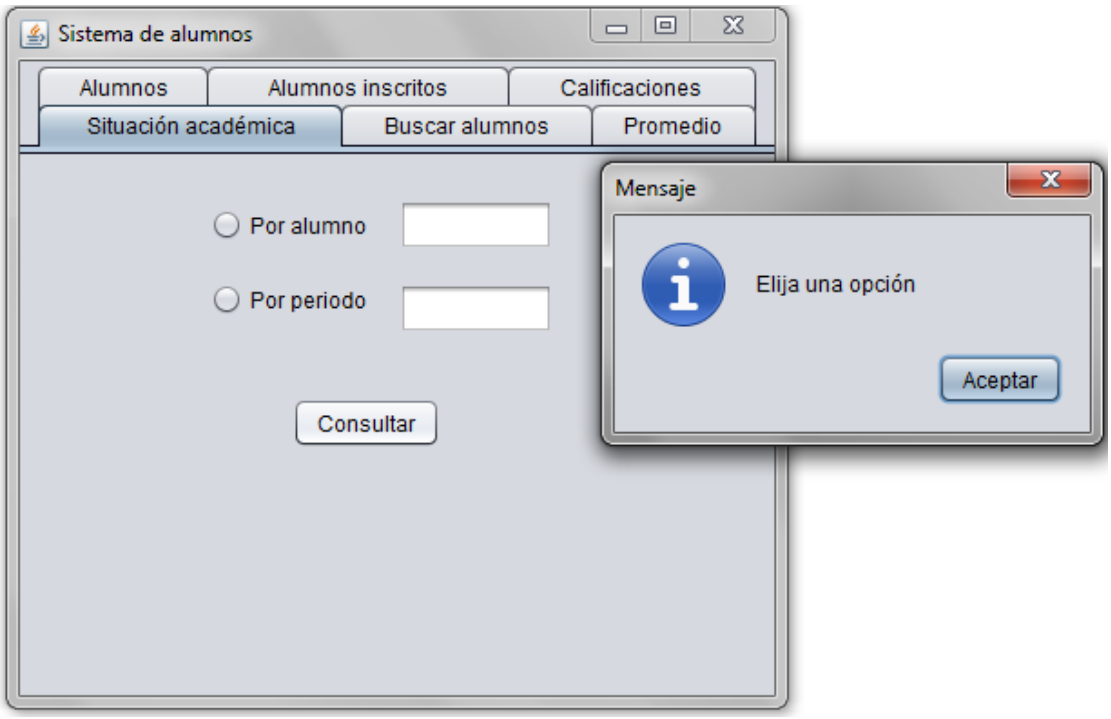


Figura 4.6: Venta de información: Elija una opción.

Capítulo 5

Conclusiones

En esta tesis se diseñó e implementó un sistema de información para seguimiento académico de alumnos de ICO del CUZ. Este sistema permite realizar consultas sobre indicadores específicos sobre de alumnos, y también del programa educativo.

Para desarrollar el sistema, se analizó la estructura del plan de estudios de la carrera, y de realizó un modelado de los datos. Esta fue una de las tareas que más consumió tiempo de desarrollo. El modelo de los datos fue traducido en las tablas principales siguientes: Alumno, Historial, Curso, Calificación y UAS. La implementación del sistema fue usando el SGBD MySQL.

Para facilitar al usuario realizar las consultas, se implementó una interfaz gráfica de usuario en lenguaje Java con la biblioteca Swing. Esta interfaz permite seleccionar la consulta requerida e ingresar los datos necesarios para producir los resultados. El sistema tiene la capacidad de identificar si faltan datos para realizar las consultas, e indicar al usuario cuáles son los datos faltantes.

Durante el diseño los problemas enfrentados fueron el determinar el tipo de dato correcto para cada atributo, así como determinar las relaciones correctas entre tablas. Es importante mencionar que hay que tener cuidado con los tipos de datos que se emplean, ya que si se eligen de manera incorrecta causan errores que simple vista no son relevantes, pero en el momento de la inserción de los datos provoca errores. Para probar el sistema se agregaron datos arbitrarios que no corresponden a alumnos reales. Se realizaron las consultas que dan información sobre indicadores del programa educativo de ICO del CUZ, y también sobre alumnos en específico. Como parte de las pruebas, se incluyeron casos en los que el usuario no introduce consultas con datos completos, y el sistema brindó recomendación en todos los casos.

Como trabajos futuros, se propone la realización de un módulo para capturar las calificaciones de usuarios reales, provenientes de archivos de texto plano que se generan en el sistema de control escolar de la UAEM. Otra propuesta de trabajo futuro, es la migración de la aplicación hacia plataforma Web con interfaces adaptables para dispositivos móviles. Para lograr lo anterior, se puede aprovechar completamente todo el modelo desarrollado.

Bibliografía

[1] V.F. Alarcón and Upc Edicions Upc. *Desarrollo de sistemas de información: una metodología basada en el modelado*. Aula Politècnica. UPC, 2006.

[2] M.V.N. Cabello. *Introduccion a Las Bases de Datos Relacionales*. Editorial Visión Libros, 1998.

[3] A. Cobo. *Diseño y programación de bases de datos*. Editorial Visión Libros, 0.

[4] C. J Date. *Introducción a los sistemas de base de datos*. Pearson Prentice Hall, 2001. isbn: 9684444192.

[5] A. de Miguel Castaño and M. G. P. Velthuis. *Fundamentos y modelos de base de datos*. Alfaomega, 1999.

[6] A.B. Durán. *Acceso a datos en aplicaciones web del entorno servidor. IFCD0210*. IC Editorial, 2015.

[7] Mark L. Gillenson. *Administración de base de datos*. Limusa Wiley, 2006. isbn-10:9681865952 isbn-13:9789681865955.

[8] T. Groussard. *JAVA 8: Los fundamentos del lenguaje Java (con ejercicios prácticos corregidos)*. Recursos Informáticos. Ediciones ENI, 2014.

[9] J.J.L. Hermoso. *Informática Aplicada a la Gestión de Empresas*. Biblioteca de economía y finanzas. Esic, 2000.

[10] J.L. Herrera. *Programación en tiempo real y bases de datos: Un enfoque práctico*. UPCGrau: Enginyeria informàtica. Universitat Politecnica de Catalunya. Iniciativa Digital Politecnica, 2011.

[11] López Quijano José. *Donime PHP y MySQL*. 2010.

[12] D.M. Kroenke and A.E.G. Hernández. *Procesamiento de bases de datos: fundamentos, diseño e implementación*. Pearson Educación, 2003.

[13] K.C. Laudon and J.P. Laudon. *Sistemas de información gerencial: administración de la empresa digital*. Pearson Educación, 2004.

[14] César Pérez López. *MySQL para Windows y Linux*. Number 978970151325. Ra-Ma, Alfaomega, 2008.

[15] C.P. López. *Minería de datos: técnicas y herramientas.* Paraninfo Cengage Learning, 2007.

[16] M. Loy and R. Eckstein. *Java Swing.* Developing GUIs in Java. O'Reilly Media, Incorporated, 2002.

[17] Steffen Macke. Dia diagram editor, 2014.

[18] M.V. Mannino. *Administración de bases de datos: diseño y desarrollo de aplicaciones.* 9701061098, 9789701061091. McGraw Hill, 3 edition, 2007.

[19] A.R. Martín and M.J.R. Martín. *Operaciones con bases de datos ofimáticas y corporativas.* Informática (Paraninfo). Thomson-Paraninfo, 2007.

[20] Rob Peter and Coronel Carlos. *Sistema de base de datos: Diseño, implementación y administración.* Thomson, quinta edition, 2004.

[21] Vianni Lily Álvarado Prado. Bases de datos espacios-temporales, 2009.

[22] R. Ramakrishnan, J. Gehrke, J.C. Fernández, A.G. Cordero, F.S. Pérez, and C.L. Martínez. *Sistemas de gestión de bases de datos.* McGraw Hillaw-Hill/Iberoamericana, 2007. isbn:9788448156381.

[23] Navathe Ramez, Elmasri; Shamkant. *Fundamentos de base de datos.* Addison-Wesley, 2007.

[24] C.M. Ricardo. *Bases de datos.* McGraw-Hill, 2009.

[25] F.L.O. Rivera. *Base de datos relacionales.* Textos académicos. 2008.

[26] I.L. Ruíz, I. Luque, M.A. Gomez-Nieto, and G.A.C. GARCIA. *Bases de datos: desde Chen hasta Codd con Oracle.* Alfaomega, Ra-Ma, 2002. isbn:9701507606.

[27] Abraham Silberschatz, Henry F. Korth, and S Sudarshan. *Fundamentos de base de datos.* McGraw Hill, 2002.

[28] Abraham Silberschatz, Henry F. Korth, and S Sudarshan. *Fundamentos de base de datos.* McGraw Hill, 2006.

[29] Alice Y. H. Tsai. *Sistemas de base de datos Administración y Uso.* Pentice Hall Hispanoamerica, 1990. isbn: 968880181x.

[30] J.D. Ullman, J. Widom, and E.A. Miguel. *Introducción a los sistemas de bases de datos.* Pearson Educación, 1999.

[31] Ignacio Vivona. Java. 2011.