

PROYECTO DE CURSO DE BASES DE DATOS

INTRODUCCIÓN

Una universidad en Bogotá D.C. ofrece 16 programas académicos en pregrado y posgrado y cuenta con equipo de unos 610 funcionarios de los cuales 80 son docentes de planta.

La Vicerrectoría de Administración y Financiera de esta institución encomendó el desarrollo de un sistema computacional soportado en una base de datos, que permita una adecuada gestión de los activos fijos tangibles de la universidad.

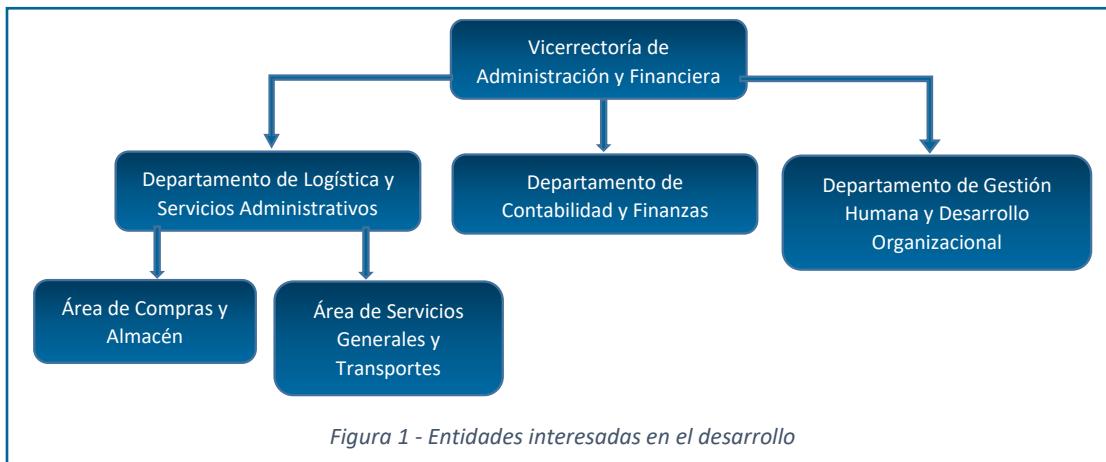
Un activo fijo tangible es un bien que tiene la universidad a su disposición para cumplir con sus funciones. Se trata de un bien que no está a la venta y cuenta con una vida útil de mediano o largo plazo.

Los activos fijos tangibles de la universidad son en su mayoría, mobiliario de las oficinas y salones, equipos electrónicos como computadores, televisores o equipos de sonido y aparatos utilizados en los laboratorios.

La gestión de estos activos es de interés para los siguientes departamentos de la universidad:

- Logística y Servicios Administrativos
- Contabilidad y Finanzas
- Gestión Humana y Desarrollo Organizacional

El siguiente organigrama resumido de la universidad, explica la relación que existe entre estas entidades:



DEFINICIÓN DEL PROBLEMA

El Área de Compras y Almacén ha venido registrando en un documento de Excel, los activos fijos que se han comprado a lo largo del año 2020. Aun no se tiene registro de todos los activos con los que contaba la universidad antes de crear este archivo. Por esta razón, se desea conformar un equipo de auxiliares de inventarios para que recorran la universidad, y vayan completando este documento.

Sin embargo, seguir utilizando el archivo Excel tiene el inconveniente de que cada vez que se realice una actualización en el mismo, el área de Compras y Almacén tendría que compartirlo con el área de Servicios

Generales y los departamentos de Contabilidad y Gestión Humana. Debido a que cada una de estas entidades se preocupa por unos datos específicos de los activos, no conviene compartir el mismo documento para todos.

Por esta razón, se desea tener una aplicación de fácil acceso y soportada en una base de datos, que contenga toda esta información, se actualice para todos los usuarios a medida que se hacen cambios e inserciones a la misma, y presente solo la información relevante para cada entidad y/o tipo de usuario.

No contar con esta herramienta en la universidad supone una serie de problemas para la universidad en términos de logística y seguridad, control contable y gestión del personal.

LOGÍSTICA Y SEGURIDAD

No hay un registro de la ubicación física de los activos de la universidad, lo cual representa un problema en términos de logística.

- *Por ejemplo, si el área de Gestión Humana desea realizar una actividad de bienestar laboral al aire libre, y solicita un conjunto de mesas, sillas, equipos de cómputo o de sonido, para organizar el evento; bastaría con que el área de Servicios Generales hiciera una consulta en el sistema para verificar cuáles de estos activos están disponibles y cuál es su ubicación, de manera que se coordine su traslado al lugar del evento.*

No contar con la ubicación física de los activos de más valor, también puede representar un problema en términos de costos y seguridad para la universidad. Saber dónde se encuentran facilitaría el proceso de vigilancia por parte del personal de la universidad dedicado a ello.

CONTROL CONTABLE

No hay información actualizada de la depreciación de los activos fijos. El departamento de Contabilidad necesita de esta información para elaborar sus informes y responder a las exigencias de la universidad y de auditorías externas.

Adicionalmente, no contar con un registro de los activos disponibles en la universidad, también puede resultar en gastos innecesarios:

- *Por ejemplo, si un funcionario necesita un reemplazo de su silla porque ya no funciona correctamente y no se puede reparar, bastaría con consultar el sistema y buscar una silla en buenas condiciones que se encuentre disponible. Esto evitaría tener que hacer una solicitud al área de Compras por una silla totalmente nueva.*

GESTIÓN DE PERSONAL

No hay un registro que indique la condición de los puestos de trabajo de los funcionarios. Algunos funcionarios pueden estar trabajando en puestos o con herramientas que presentan fallas físicas y técnicas, reduciendo su comodidad y eficiencia, y en algunos casos hasta afectando su salud sin siquiera saberlo. Esta información es de utilidad para el equipo encargado de la salud en el trabajo de la universidad.

Por otro lado algunos funcionarios terminan su contrato de trabajo y se retiran de la universidad, pero no se sigue un proceso de devolución del puesto de trabajo con la debida firma de paz y salvo. Contar con un sistema que registre los activos que están bajo la responsabilidad de un funcionario facilitaría este proceso, y permitiría imponer alguna sanción (si aplica) cuando se evidencie un daño en el activo fijo al momento de su devolución.

OBJETIVO Y CARACTERÍSTICAS DEL PRODUCTO FINAL

Desarrollar una aplicación web construida a partir de una base de datos de los activos fijos tangibles de la universidad, de manera que:

- presente información relevante para cada una de las áreas interesadas,
- soporte la gestión de este inventario y
- sirva de registro de una serie de procedimientos inherentes a esta gestión

A continuación se presenta la información que se debe presentar en la aplicación según cada tipo de usuario

Información relevante para cada tipo de usuario según la entidad para la que trabaja			
Usuario	Entidad	Información	Permiso
Auxiliar de Inventarios	Área de Compras y Almacén	La descripción de los activos fijos, incluyendo sus datos contables y de ubicación física.	Permiso para insertar y actualizar registros. <i>Es necesario a medida que realizan el inventario de los activos existentes o se adquiere un nuevo activo desde la oficina de Compras.</i>
Auxiliar de Mantenimiento	Área de Servicios Generales y Transportes	La descripción de los activos fijos que necesitan una reparación o mantenimiento, incluyendo su ubicación física.	Permiso para consultar. <i>Conocer la descripción física y ubicación de los activos, permitirá identificarlos y realizar las reparaciones o mantenimientos necesarios.</i>
Personal de Servicios Generales y Transportes	Área de Servicios Generales y Transportes	La descripción de los activos fijos, incluyendo su ubicación física.	Permiso para consultar. <i>Conocer la descripción física y ubicación de los activos, permitirá coordinar traslados cuando se necesiten y organizar eventos</i>
Personal de Contabilidad y Finanzas	Departamento de Contabilidad y Finanzas	Los datos de identificación de los activos fijos sin su descripción física e incluyendo los datos contables.	Permiso para actualizar y eliminar registros. <i>Es posible que esta área decida que un registro no entre dentro de la categoría de activo fijo para la universidad y proceda a eliminarlo. También pueden decidir ajustar los índices de depreciación de ciertos activos.</i>
Personal de Gestión Humana y Desarrollo Organizacional	Departamento de Gestión Humana y Desarrollo Organizacional	La descripción de los activos fijos incluyendo su descripción física de los activos asignados a los funcionarios de la universidad, incluyendo detalles sobre su estado según la última revisión realizada por los auxiliares de inventarios.	Permiso para consultar. <i>Así podrán tomar decisiones para preservar la comodidad, salud y seguridad de los funcionarios en sus puestos de trabajo, como imponer sanciones cuando se evidencie algún daño intencionado o por negligencia de parte de un funcionario a su puesto o herramientas de trabajo.</i>

Tabla 1 - Información relevante para cada entidad interesada

Adicional a estas características, la aplicación debe permitir a los auxiliares de inventarios, llevar registro de cuatro procedimientos relacionados con la gestión de los activos fijos de la universidad, y que permitirán responder a las necesidades de las entidades interesadas.

Revisión: Cuando el Departamento de Logística y Servicios Administrativos recibe una notificación de parte de algún área o funcionario de la universidad en la que se solicita una revisión de su puesto de trabajo o herramienta porque presenta alguna falla, un auxiliar de inventarios se acerca al lugar y realiza una verificación o diagnóstico. El auxiliar de inventarios decide si se debe solicitar la compra (reemplazo) del activo o si se debe coordinar cuna cita de mantenimiento con el personal de Servicios Generales. Los auxiliares de inventarios realizan rondas cada cierto verificando la consistencia de la base de datos. Cuando encuentran alguna anomalía, pueden registrarla con este tipo de procedimiento.

Entrega: Cuando un funcionario es contratado en la universidad, un auxiliar de inventarios se acerca al puesto de trabajo y formaliza la entrega y asignación del mobiliario, equipo y herramientas que requiere el trabajador. Se hace el registro del proceso y se asigna el funcionario en cuestión a cada uno de los activos. Es la entrega formal de su puesto de trabajo, el cual en la mayoría de los casos consiste de una silla, un escritorio y un computador. Algunos funcionarios tendrán a su disposición otros dispositivos y herramientas según las funciones que desempeñe. Este procedimiento es de interés para el departamento de Gestión Humana, con el fin de llevar un registro de las condiciones de los puestos de trabajo y herramientas de sus funcionarios, de manera que puedan prevenir enfermedades y accidentes laborales, y realizar análisis del impacto que tiene el ambiente de trabajo en el desempeño de los trabajadores. Al realizar este procedimiento, se busca actualizar el registro de la tabla ACTIVO indicando el funcionario al cuál se le está haciendo la entrega. Este procedimiento es realizado por un auxiliar de inventarios.

Paz y salvo: También llamado devolución del puesto de trabajo. Cuando un funcionario se retira de la compañía, un auxiliar de inventarios se acerca al puesto de trabajo y revisa el estado de los activos que lo componen. Toma las observaciones necesarias y se determina si se puede o no firmar su paz y salvo en coordinación con el personal de Gestión Humana. El activo puede estar en buenas condiciones, necesitar un mantenimiento de parte de los auxiliares de mantenimiento o requerir ser reemplazado por uno nuevo. Los auxiliares de mantenimiento tienen acceso a la información de los activos que requieren mantenimiento y se acercan al lugar a cumplir con esta función. Asimismo se debe retirar el funcionario responsable de cada uno de los registros de los activos en cuestión, es decir formalizar la devolución del puesto de trabajo.

Mantenimiento: Los auxiliares de mantenimiento registran las citas de mantenimiento completadas, para cada uno de los activos reparados. Hay un espacio para escribir las observaciones del proceso. La lista de activos que necesitan ser reemplazados se actualiza en una tabla visible para el área de Compras y Almacén.

FASES DEL PROYECTO U OBJETIVOS ESPECÍFICOS

Las fases a seguir para alcanzar el objetivo encomendado son:

DISEÑO

Diseñar la estructura de la base de datos para los activos fijos. La base de datos debe ser consistente con otros sistemas ya existentes en la universidad, como es el de administración del personal, donde se encuentra el registro de todos los funcionarios de la universidad según su posición en el organigrama.

IMPLEMENTACIÓN

Implementar la estructura de la base de datos en un motor de bases de datos (Oracle APEX) haciendo la respectiva inserción de datos, según la información disponible hasta el momento: activos fijos registrados en el archivo Excel hasta el momento, datos de los funcionarios, estructura organizacional y física de la universidad.

DESARROLLO

Desarrollar la aplicación web con el constructor de aplicaciones del motor utilizado y soportada en la base de datos ya implementada, incluyendo todas las funcionalidades y características mencionadas anteriormente en el documento.

DISEÑO DE LA BASE DE DATOS

Especificaciones del diseño

La tabla más importante de la base de datos es ACTIVO, la cual contiene una descripción física de cada activo fijo, junto con su ubicación física y datos contables.

Algunos activos fijos son asignados a un funcionario responsable por medio del atributo ID_FUNCIONARIO, el cual es una llave foránea que relaciona a la tabla ACTIVO con la tabla FUNCIONARIO.

La universidad cuenta con un registro de todos los funcionarios de la empresa actualmente, que suman 610, de los cuales 80 son docentes de planta. No se incluyen en esta cuenta a los directores de departamento, decanos de facultad ni vicerrectores.

La base de datos a diseñar, debe ser consistente con otros sistemas computacionales de la universidad, y por esta razón, se debe representar el esquema organizacional de la compañía en la base de datos. A continuación los detalles:

La universidad es dirigida por cuatro vicerrectorías. A continuación se presentan, junto con el vicerrector asignado actualmente.

- Vicerrectoría de Formación: María Claudia Rincón Mora
- Vicerrectoría de Administración y Financiera: Luz Marina Trejos Montero
- Vicerrectoría de Investigación: Carlos Humberto Angarita Preciado
- Vicerrectoría de Extensión y Proyección Social: Luis Augusto Moreno Boadas

Adicionalmente cuenta con cinco facultades. Cada facultad tiene un decano asignado.

- Facultad de Ciencias Económicas, Administrativas y Contables: Mario Alberto Rodríguez Franco
- Facultad de Ciencias Jurídicas y Sociales: Francisco Antonio Pedraza Vásquez
- Facultad de Ciencias Agrarias: Néstor Mauricio Serna Rodríguez
- Facultad de Ingeniería: María Luisa Pérez Cifuentes
- Facultad de Ciencias Básicas y de la Educación: Sandra Milena Piedrahita Buitrago

La universidad cuenta con 23 departamentos, los cuales pueden pertenecer a una vicerrectoría o a una facultad, pero no a ambos al mismo tiempo. Cada departamento tiene un director asignado.

Vicerrectoría	Departamento	Director
Formación	Formación Académica	Carlos Andrés Preciado Sosa
	Educación Virtual	Carlos Arturo Durán Porras
	Aseguramiento de la Calidad	Mario Andrés Canadas Buitrago
Administración y Financiera	Logística y Servicios Administrativos	Wilson Andrés Montaño Cárdenas
	Administración Dependencia Teusaquillo	María Alejandra Azuero López
	Administración Dependencia Facatativá	Beatriz Gaitán Oviedo
	Mercadeo	Johanna Vanessa Duarte Jiménez
	Secretaría General	María Helena Cadavid Quintana
	Sistemas y Tecnología	Daniel Humberto Díaz Calero
	Gestión Humana y Desarrollo Organizacional	María Alejandra Pinzón Ramos
	Contabilidad y Finanzas	Carlos Alberto Bernal Tiquiza
	Planeación	David Hernando Torres Piraquive
Investigación	Investigación Formativa	Adriana Martínez Méndez
	Investigación Aplicada	Pablo Andrés Rodríguez Peña
Extensión y Proyección Social	Bienestar Universitario	Eduardo Fernández Wilches
	Proyectos Educativos en Contextos Rurales	Sandra Patricia Lara Martínez
	Educación para el Trabajo y el Desarrollo Humano	Samuel Cipriano Moreno Rodríguez

Tabla 2 - Dominio: Directores de departamento por vicerrectoría

Facultad	Departamento	Director
Ciencias Agrarias	Investigación en Ciencias Agrarias	Manuel Carlos Medina Lopera
Ciencias Básicas y de la Educación	Humanidades	Martín Alejandro Corrales Parra
	Investigación en Ciencias Básicas	Diego Alonso Mancilla Perilla
	Idiomas	Roger Miller Campbell
	Ciencias Básicas	Myriam Adriana Rodríguez López
Ingeniería	Investigación en Ingeniería	Gloria Mercedes Cardona Sotelo

Tabla 3 – Dominio: Directores de departamento por facultad

Un nivel más abajo en el organigrama, la universidad cuenta con 56 áreas, las cuales pueden pertenecer a una facultad o a un departamento, pero no a ambos.

Facultad	Área
Ciencias Agrarias	Jefatura del CIDT Pinares de Tenjo
	Sección de Laboratorios de Ciencias Agrarias
Ciencias Básicas y de la Educación	Sección de Laboratorios de Ciencias Básicas
	Escuela de Formación y Perfeccionamiento Docente
Ciencias Económicas, Administrativas y Contables	Consultorio Contable
Ingeniería	Sección de Laboratorios de Ingeniería
Ciencias Jurídicas y Sociales	Consultorio Jurídico
	Centros de Conciliación

Tabla 4 - Dominio: Áreas por facultad

Departamento	Área
Investigación Aplicada	Ética y Bioética
Investigación Formativa	Divulgación Científica

Aseguramiento de la Calidad	Programa Sembrar Paz Autoevaluación Registros Calificados Acreditación
Proyectos Educativos en Contextos Rurales	Educación Permanente Articulación con la Educación Media
Gestión Humana y Desarrollo Organizacional	Capacitación y Desarrollo de Personal Seguridad y Salud en el Trabajo Reclutamiento y Selección de Personal Administración de Personal
Logística y Servicios Administrativos	Servicios Generales y Transportes Compras y Almacén Infraestructura
Sistemas y Tecnología	Soporte Sistema Institucional Soporte Técnico y Telecomunicaciones Ayudas Educativas
Administración Dependencia Facatativá	Servicios Administrativos Servicios del Medio Universitario Jefatura de Ingeniería Civil Jefatura de Ingeniería Mecatrónica Jefatura de Derecho
Administración Dependencia Teusaquillo	Servicios Administrativos Servicios del Medio Universitario Jefatura de Derecho
Educación Virtual	Innovación Educativa Desarrollo de Programas Virtuales Ambientes Virtuales de Aprendizaje
Bienestar Universitario	Acompañamiento Estudiantil Integración con el Egresado Psicología Clínica Odontología Arte y Cultura Actividad Física y Deporte Medicina Estudiantil
Formación Académica	Emprendimiento Estudios Ambientales Registro y Control Sección de Bibliotecas
Secretaría General	Archivo y Gestión Documental
Planeación	Sistemas Integrados de Gestión Gestión de la Información Institucional Planeación Presupuestal Planificación Estratégica
Mercadeo	Permanencia Estudiantil Comunicaciones Investigación de Mercados

Tabla 5 - Dominio: Áreas por departamento

Cada área cuenta con un jefe de área, dos coordinadores, cuatro asistentes y dos asistentes junior o aprendices.

Es decir, se tienen las siguientes cifras

- 56 jefes de área: un jefe por área
- 112 coordinadores: dos por área
- 224 asistentes: cuatro por área

- 112 asistentes junior: dos por área

Lo cual suma un total de 504 empleados en el sector administrativo de la universidad.

Adicionalmente se cuenta con 26 auxiliares:

- 3 auxiliares de inventarios – Área de Compras y Almacén
- 3 auxiliares de mantenimiento – Área de Servicios Generales y Transportes
- 12 auxiliares de laboratorio – Sección de Laboratorios de Ingeniería
- 5 auxiliares de laboratorio – Sección de Laboratorios de Ciencias Agrarias
- 3 auxiliares de laboratorio – Sección de Laboratorios de Ciencias Básicas

La universidad ofrece 16 programas académicos actualmente. Cada programa pertenece a una facultad:

Facultad	Programa
Ciencias Agrarias	Especialización en Bienestar Animal y Etología
	Medicina Veterinaria
	Zootecnia
Ciencias Básicas y de la Educación	Licenciatura en Ciencias Naturales y Educación Ambiental
Ciencias Económicas, Administrativas y Contables	Administración Financiera y de Sistemas
	Contaduría Pública
	Especialización en Gestión de Agronegocios
	Especialización en Sistemas de Gestión Integrada HSEQ
Ingeniería	Especialización en Seguridad Industrial, Higiene y Gestión Ambiental
	Ingeniería Agroindustrial
	Ingeniería Civil
	Ingeniería de Alimentos
	Ingeniería Industrial
	Ingeniería Mecatrónica
Ciencias Jurídicas y Sociales	Derecho
	Especialización en Legislación Rural y Ordenamiento Territorial

Tabla 6 - Dominio: Programas de la universidad

Actualmente la universidad cuenta con un plantel de 80 docentes cuyo contrato es a tiempo completo, tienen funciones administrativas en sus programas y facultades y trabajan en investigación.

Los docentes están adscritos a un programa (en lugar de un área) y además cuentan con un rol entre cuatro escalafones que maneja la universidad.

Nombre del programa	Rol / Escalafón	Nº
Administración Financiera y de Sistemas	Asociado	2
	Instructor	2
Contaduría Pública	Titular	2
	Instructor	2
	Asociado	2
Derecho	Titular	3
	Instructor	2
	Asociado	3
Especialización en Bienestar Animal y Etología	Titular	2
	Instructor	1

	Asistente	1
Especialización en Gestión de Agronegocios	Asociado	1
	Titular	1
	Instructor	1
	Instructor	1
Especialización en Legislación Rural y Ordenamiento Territorial	Asociado	2
	Titular	2
	Instructor	1
Especialización en Seguridad Industrial, Higiene y Gestión Ambiental	Asociado	1
	Instructor	1
	Titular	1
Especialización en Sistemas de Gestión Integrada HSEQ	Titular	3
	Instructor	2
	Asistente	2
Ingeniería Agroindustrial	Instructor	2
	Titular	3
	Asistente	2
Ingeniería Civil	Asistente	1
	Instructor	1
	Titular	2
Ingeniería de Alimentos	Asociado	1
	Instructor	1
	Titular	1
Ingeniería Industrial	Asistente	1
	Instructor	1
	Titular	1
Ingeniería Mecatrónica	Instructor	3
	Asociado	3
	Titular	3
Licenciatura en Ciencias Naturales y Educación Ambiental	Asociado	3
	Titular	3
	Instructor	2
Medicina Veterinaria	Asociado	1
	Instructor	1
	Titular	1
Zootecnia	Asociado	2
	Instructor	1

Tabla 7 - Número de docentes por rol para cada programa de la universidad.

El resultado obtenido fue aleatorio según la inserción de datos que se realizó posteriormente en la base de datos

Cada docente tiene un título que representa el máximo nivel académico alcanzado en sus estudios. En general, todos los docentes con el rol de Instructor tienen un título de Especialización. Los docentes con un título de Maestría pueden pertenecer a las categorías de Asistente, Asociado y Titular. Los docentes con un título de Doctorado son docentes asociados y titulares.

Así entonces, la jerarquía de entidades en la universidad está dada por el siguiente diagrama:

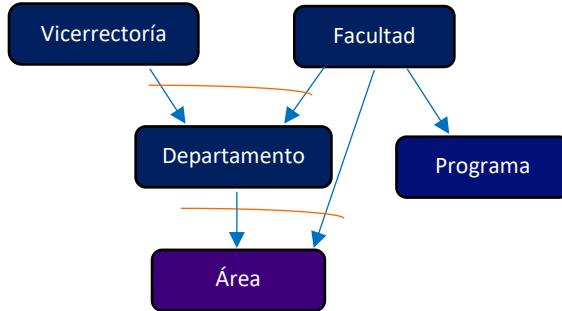


Figura 2 – Relaciones jerárquicas entre las entidades de la universidad

Los arcos en color naranja indican una relación opcional excluyente.

En relación a la jerarquía en los cargos de la universidad. Se tiene que los vicerrectores están en el nivel más alto junto con los decanos de facultad. Un nivel más abajo está los directores de departamento quienes pueden trabajar para una facultad o una vicerrectoría. Vicerrectores, decanos y directores no están siendo contados dentro de los 610 funcionarios de la empresa.

Este grupo de trabajadores en la empresa no son interesantes en la asignación de activos fijos, y por eso no necesitan incluirse dentro de la tabla que contenga a todos los funcionarios de la universidad.

El siguiente diagrama describe las relaciones de jerarquía en la universidad:

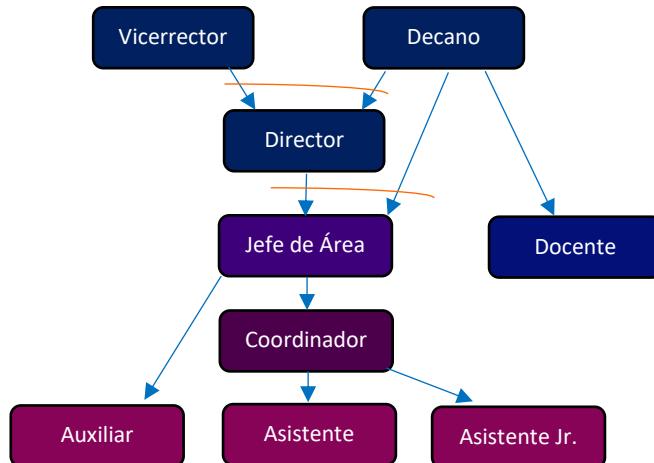


Figura 3 – Relaciones jerárquicas entre las entidades de la universidad incluyendo los cargos de los funcionarios

Los arcos en color naranja indican una relación opcional excluyente. Los auxiliares tienen como jefe inmediato al jefe de área directamente. Algunos jefes de área son subordinados de un Decano y otros de un Director, tal como se mostró en la tabla de Áreas anteriormente.

Entre los docentes no hay una jerarquía diferente de su escalafón. Todos responden al decano de la facultad del programa al que pertenecen.

Adicional a esta estructura organizacional descrita, la base de datos debe reflejar la estructura física de la universidad según otros sistemas actuales de la universidad, con el fin de guardar la consistencia que se necesita y para registrar la ubicación física de los activos.

La universidad está dividida en cuatro sedes:

- Sede Principal
- Dependencia Facatativá
- Dependencia Teusaquillo
- CIDT Pinares de Tenjo

La sede más grande es la principal, y como tal, tiene un mayor número de edificios o bloques. A continuación se listan los edificios que pertenecen a la universidad para cada una de las sedes:

Sede	Edificio
Sede Principal	Bloque A
	Bloque B
	Bloque C
	Bloque D
	Bloque E
	Bloque F
	Bloque G
	Bloque H
	Bloque I
	Bloque J
Dependencia Facatativá	Bloque Principal
	Bloque Carrera 15
Dependencia Teusaquillo	Bloque Principal
	Bloque Carlos Fernando Paredes Millán
CIDT Pinares de Tenjo	Bloque Principal
	Bloque Administrativo
	Bloque Laboratorios
	Bloque Operarios

Tabla 8 - Dominio: Edificios de la universidad

La universidad tiene a su disposición 544 salones de clase repartidos en cada sede, así:

- Sede Principal: 262 salones
- Dependencia Teusaquillo: 92 salones
- Dependencia Facatativá: 78 salones
- CIDT Pinares de Tenjo: 112 salones

De entre los cuales hay 19 talleres, en los cuales se realizan actividades extracurriculares como danza, música, carpintería y de algunas asignaturas electivas de los programas como fotografía, pintura, etc.

Dentro de estos 544 salones también hay 14 salas de cómputo, cuya capacidad está dada por el número de computadores disponibles en dicha locación.

Además, hay 9 auditorios en total repartidos en algunos edificios de la universidad:

Sede	Edificio	Auditorio	Capacidad
Sede Principal	Bloque A	Auditorio Santiago Francisco Mora Guerra	120
	Bloque C	Auditorio Manuel Rafael Salas Gil	100
	Bloque E	Auditorio Orlando Jerónimo Varela Salazar	130
	Bloque G	Auditorio Alejandra Silvana Pinilla Vásquez	100
	Bloque I	Auditorio Ricardo Felipe Romero Ortega	130
Dependencia Teusaquillo	Bloque Principal	Auditorio John Orlando Galeano Moreira	130
Dependencia Facatativá	Bloque Principal	Auditorio Jéssica Verónica Caballero Rivera	130

CIDT Pinares de Tenjo	Bloque Operarios	Auditorio Andrea Gina Vásquez Prieto	110
	Bloque Principal	Auditorio Rosa Johanna Costa Gómez	130

Tabla 9 - Dominio: Auditórios de la universidad

El resultado de los nombres para cada auditorio como de su capacidad fue obtenido aleatoriamente del procedo de inserción de datos que se describe más adelante en el documento.

Otra parte importante de la estructura física de la universidad son las oficinas. En total, en toda la universidad hay 84 oficinas, algunas lo suficientemente grandes como para contener hasta diez puestos de trabajo o cubículos. Las más pequeñas tienen solamente tres puestos de trabajo.

La repartición de las oficinas por sede es la siguiente:

- Sede Principal: 44 oficinas
- Dependencia Teusaquillo: 13 oficinas
- Dependencia Facatativá: 11 oficinas
- CIDT Pinares de Tenjo: 16 oficinas

Cada oficina pertenece a un área en específico. Un área puede tener 1 o 2 oficinas para sus trabajadores.

Finalmente, se tiene un tipo especial de locación, el cual corresponde a los laboratorios. En total hay 19 laboratorios en la universidad. 15 de ellos se encuentran en la sede principal y 4 en el CIDT Pinares de Tenjo. Cada laboratorio tiene asignado a un auxiliar, quien se responsabiliza por una serie de tareas operativas encomendadas desde la sección de laboratorios a la que esté adscrito.

Sede	Edificio	Laboratorio	Auxiliar	Área
Sede Principal	Bloque E	Biocombustibles	Steven José Sánchez Silva	Sección de Laboratorios de Ingeniería
		Investigación, Calidad y Análisis	Ángel Paul Beltrán López	
		Investigación y Desarrollo Metrológico	Jaime Andrés Prieto Asensio	
		Instrumentación Analítica	Edgar Eloy Santos Montero	
		Química	Julián José Santamaría Varela	Sección de Laboratorios de Ciencias Básicas
		Ciencias Biológicas	Nathaly Myriam Montes Campos	
		Física	Néstor Paul Usoaquén Pérez	
		Fitoquímica	Silvia Nicol Segura Herrera	Sección de Laboratorios de Ciencias Agrarias
	Bloque I	Materiales	Victor Augusto Galeano Núñez	
		Neumática	Alex Bernardo Valencia Flores	
		Ingredientes Naturales	Valentina Elisa Rey Peña	
		Hidráulica	Nicol Gina Molina Álvarez	
		Bioensayos y Fuentes de Alimentación No Convencionales	Alexis David Valencia Ramírez	
		Electrónica	Lorena Luz Espinoza Salas	
		Geomática	César Andrés Rubio Murillo	
CIDT Pinares de Tenjo	Bloque Laboratorios	Mecánica de Suelos	Valentina Lina Amador Peña	Sección de Laboratorios de Ciencias Agrarias
		Microbiología	Alberto Eloy Calderón Domínguez	
		Nutrición y Alimentación	Mónica Xiomara Segura Caballero	Sección de Laboratorios de Ingeniería
		Pavimentos	Jerónimo Daniel Cabrera Arias	

Tabla 10 - Dominio: Laboratorios de la universidad y Auxiliares de laboratorios.

Los nombres para cada auxiliar se generaron aleatoriamente según el proceso de inserción

Hay un auxiliar de laboratorio que aún no ha sido asignado a ningún laboratorio. Se trata de Enrique Edison Cardona Rey que trabaja para la Sección de Laboratorios de Ciencias Agrarias.

Así entonces, la estructura física de la universidad está dada por el siguiente diagrama:

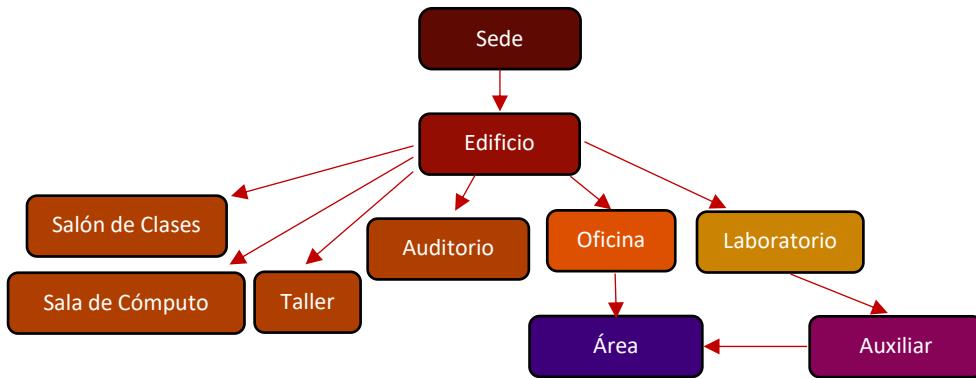


Figura 4 – Diagrama que detalla la estructura física de la universidad y su relación con algunas entidades

Finalmente se hablará de los activos en sí. Debe existir un registro por activo en la base de datos. Este registro debe indicar los datos de identificación del mismo, detalles sobre su descripción física y otros que aparezcan en la factura de compra, la información de interés para el personal de Contabilidad, su ubicación física y funcionario responsable, cuando aplique.

Habiendo definido todas estas especificaciones, están las condiciones dadas para iniciar la metodología de diseño de la base de datos.

Metodología de diseño de la base de datos

Definición de entidades

Inicialmente se consideraron

- ACTIVO
- COMPRA
- FUNCIONARIO
- ÁREA
- DEPARTAMENTO
- FACULTAD
- PROGRAMA
- VICERRECTORÍA
- LOCACIÓN
- EDIFICIO
- SEDE

Definición de subtipos y supertipos

Hay dos supertipos en las entidades listadas anteriormente: FUNCIONARIO y LOCACIÓN.

Un funcionario puede ser un EMPLEADO o un DOCENTE.

Los empleados son los funcionarios que hacen parte del plantel administrativo u operativo de la universidad. Dentro de esta categoría solo se están incluyendo los jefes de área, coordinadores, asistentes y auxiliares

guardando la jerarquía presentada en la figura [3]. No se incluirán vicerrectores, directores ni decanos. Los empleados están adscritos a un área específica de la universidad.

Se considerará otro subtipo llamado AUXILIAR, el cual es una especialización de EMPLEADO.

Los docentes conforman el plantel contratado en la universidad para ejercer su labor en tiempo completo, apoyar en tareas administrativas y de investigación. Están adscritos a un programa. No existe una jerarquía entre ellos diferente a su escalafón.

Una locación puede ser un SALÓN, una OFICINA o un LABORATORIO, según se muestra en la figura [4]. Los auditorios y salones de clase se considerarán como salones, es decir, como un único subtipo.

Más adelante se presentarán los conjuntos de atributos de cada una de estas entidades, que mostrarán que si se pueden hacer estas especializaciones, según los dos criterios:

- Los subtipos tienen una llave primaria común que permanecerá en su supertipo
- Los subtipos tienen un subconjunto de atributos en común que permanecerá en su supertipo

Así entonces, las entidades definitivas para los diagramas de entidad-relación son las siguientes:

- E1 – ACTIVO
- E2 – COMPRA
- E3 – FUNCIONARIO
- E4 – DOCENTE
- E5 – EMPLEADO
- E6 – AUXILIAR
- E7 – VICERRECTORÍA
- E8 – DEPARTAMENTO
- E9 – ÁREA
- E10 – FACULTAD
- E11 – PROGRAMA
- E12 – SEDE
- E13 – EDIFICIO
- E14 – LOCACION
- E15 – OFICINA
- E16 – SALÓN
- E17 – LABORATORIO

Definición de relaciones entre entidades

Las relaciones entre entidades, incluyendo los subtipos generados, se presentan en la siguiente tabla:

RELACIONES FINALES		NOMBRE	TIPO	CARDINALIDAD	PARTICIPACIÓN
E1 - E3	ACTIVO - FUNCIONARIO	USADO-POR		m-1	> <
E1 - E6	ACTIVO - AUXILIAR	PROCEDIMIENTO		m-m	> <

E1 - E14	ACTIVO - LOCACIÓN	<i>SE-ENCUENTA-EN</i>		m-1	>
E3 - E3	FUNCIONARIO - FUNCIONARIO	<i>TRABAJA-PARA</i>	RECURSIVA	m-1	>
E6 - E17	AUXILIAR - LABORATORIO	<i>AUXILIAR-LAB</i>		1-1	... -
E4 - E11	DOCENTE - PROGRAMA	<i>TRABAJA-EN</i>		m-1	> - ...
E5 - E9	EMPLEADO - ÁREA	<i>TRABAJA-EN</i>		m-1	> - ...
E9 - E15	ÁREA - OFICINA	<i>área-oficina</i>		1-m	- - <
E9 - E10	ÁREA - FACULTAD	<i>área-facultad</i>	ARCO	m-1	>
E9 - E8	ÁREA - DEPARTAMENTO	<i>área-departamento</i>		m-1	>
E13 - E12	EDIFICIO - SEDE	<i>edificio-sede</i>		m-1	> - ...
E14 - E13	LOCACIÓN - EDIFICIO	<i>locación-edificio</i>	DEPENDENCIA	m-1	> - ...
E11 - E10	PROGRAMA - FACULTAD	<i>programa-facultad</i>	DEPENDENCIA	m-1	> - ...
E8 - E7	DEPARTAMENTO - VICERRECTORIA	<i>departamento-vicerrectoría</i>	ARCO	m-1	>
E8 - E10	DEPARTAMENTO - FACULTAD	<i>departamento-facultad</i>		m-1	>
E1 - E2	ACTIVO - COMPRA	<i>activo-compra</i>		m-1	> - -

Tabla 11 - Relaciones entre entidades según la metodología de diseño E-R

Relación m-m AUXILIAR-ACTIVO

Hay una relación m-m que corresponde a AUXILIAR-ACTIVO y se refiere particularmente a los procedimientos que deben llevar a cabo los auxiliares de inventarios y de mantenimiento, es decir, las entregas de puestos de trabajo, las firmas de paz y salvo cuando un funcionario se retira y devuelve su puesto, las revisiones para verificar el estado de los activos y los mantenimientos cuando se necesita de alguna reparación.

Relación recursiva FUNCIONARIO-FUNCIONARIO

De acuerdo con la jerarquía mostrada en la figura [3], un asistente tiene como jefe a un coordinador, un coordinador tiene como jefe a un jefe de área. Los superiores de los jefes de área no serán registrados, y se tomarán como la cabeza del árbol. Los auxiliares estarán supervisados por el jefe de su área directamente.

Relaciones opcionales excluyentes entre sí o ARCOS

Hay dos arcos presentes en el modelo. Un área puede pertenecer a un departamento o a una facultad, pero no a ambos al mismo tiempo. Un departamento puede pertenecer a una vicerrectoría o a una facultad, pero no a ambos al mismo tiempo.

Relaciones de dependencia para generar llaves primarias compuestas

En el modelo hay dos relaciones débiles: LOCACION y PROGRAMA

De acuerdo con [1], una entidad débil es aquella que depende existencialmente de una entidad identificadora. No cuenta con una llave primaria dentro de su esquema, sino con un atributo llamado discriminante el cual es único entre el conjunto de entidades débiles que se relacionan con una misma entidad identificadora, pero puede llegar a repetirse si se compara con entidades débiles relacionadas a entidades identificadoras diferentes. La llave primaria de esta entidad es la concatenación entre la llave primaria de su entidad identificadora y el discriminante.

Para solucionar esta circunstancia, se hace uso de una relación de dependencia de la entidad débil a la entidad identificadora. A continuación se presentan las dos relaciones de este tipo que se tienen en el modelo.

LOCACIÓN-EDIFICIO: Cada edificio tiene un identificador para cada locación, pero estos identificadores pueden repetirse entre edificios.

PROGRAMA-FACULTAD: Cada facultad tiene un identificador para cada programa, pero estos identificadores pueden repetirse entre facultades.

Con estas dos relaciones de dependencia se genera una llave compuesta en las entidades LOCACION y PROGRAMA.

A continuación se presenta la matriz de relaciones inicial.

MATRIZ DE RELACIONES																	
	E1	E2	E3	E4	E5	E6	E7	E8	E9	E10	E11	E12	E13	E14	E15	E16	E17
E1		m-1	m-1				m-m								m-1		
E2																	
E3			m-1														
E4													m-1				
E5									m-1								
E6																1-1	
E7										m-1							
E8									1-m	m-1							
E9										m-1					1-m		
E10										1-m							
E11																	
E12												1-m					
E13													1-m				
E14																	
E15																	
E16																	
E17																	

Figura 5 – Matriz de relaciones inicial según la metodología de diseño E-R

Definición de atributos para cada una de las entidades

ENTIDAD		ATRIBUTO	TIPO DE DATO	DOMINIO E INTEGRIDAD POR VALOR	EJEMPLO
E1	ACTIVO	# id-activo	Integer	Enteros positivos mayores o iguales a 1	1294
		* referencia-activo	Varchar(40)	Códigos alfanuméricos de hasta 40 caracteres	'XE8IBEYEFPDF6FWZ S6K5'
		* categoría-activo	Char(3)	Integridad por valor {CAE, DTS, LAB, LOG, SGT}	'DTS'
		* nombre-activo	Varchar(40)	Dado por el tipo de dato Varchar(40)	'Computador de Escritorio'
		o material-activo	Varchar(20)	Conjunto de materiales posibles	'Cuero'
		* marca-activo	Varchar(25)	Conjunto de marcas posibles	'Cecotti'
		o color-activo	Varchar(20)	Conjunto de colores posibles	'Azul Oscuro'
		o tamaño-activo	Varchar(20)	Adjetivos de tamaño, dimensiones en alguna unidad, bidimensionales y tridimensionales	'25 pulgadas'
		o componentes-activo	Varchar(120)	Dado por el tipo de dato Varchar(120)	'Soporte, Kit de Repuesto'
		* valor-activo	Money(16, 2)	Reales positivos hasta 99999999999999.99 con dos cifras decimales de precisión	4003603

		* depreciación-activo	Porcentaje	Reales positivos hasta 100 con dos cifras decimales de precisión	8
E2	COMPRA	# id-compra	Integer	Enteros positivos mayores o iguales a 2020000	2020017
		* fecha-compra	Date	Fechas de la forma 'DD/MM/YYYY' desde 01/01/2020	9/02/2020
		* valor-compra	Money(16, 2)	Reales positivos hasta 99999999999999.99 con dos cifras decimales de precisión	16084234
E3	FUNCIONARIO	# id-funcionario	Integer	Enteros positivos mayores o iguales a 1017005000	1018005241
		* nombre-funcionario	VARCHAR(50)	Dado por el tipo de dato Varchar(50)	'Sergio Emilio Molina Pereira'
		* ini_contrato-funcionario	Date	Fechas de la forma 'DD/MM/YYYY' desde 01/01/2020	1/04/2020
		o fin_contrato-funcionario	Date	Fechas de la forma 'DD/MM/YYYY' desde 31/01/2020	30/06/2020
E4	DOCENTE	* título-docente	VARCHAR(100)	{'Especialización', 'Maestría', 'Doctorado'}	'Especialización'
		* rol-docente	VARCHAR(20)	{'Instructor', 'Asistente', 'Titular', 'Asociado'}	'Asociado'
E5	EMPLEADO	* cargo-empleado	VARCHAR(30)	{'Jefe de Área', 'Coordinador', 'Asistente', 'Asistente Jr.', 'Auxiliar'}	'Asistente Jr.'
E6	AUXILIAR	* tipo-auxiliar	CHAR(3)	Integridad por valor {INV, MAN, LAB}	'INV'
E7	VICERRECTORÍA	# id-vicerrectoría	Char(3)	Integridad por valor {FOR, INV, EXT, ADM}	'EXT'
		* nombre-vicerrectoría	VARCHAR(50)	Nombres de las 4 vicerrectorías existentes	'Vicerrectoría de Administración y Financiera'
		* vicerrector-vicerrectoría	VARCHAR(50)	Nombres de los 4 vicerrectores existentes	'Luis Augusto Moreno Boadas'
E8	DEPARTAMENTO	# id-departamento	Integer	Entero entre 1 y 23 incluyendo extremos	21
		* nombre-departamento	VARCHAR(70)	Nombres de los 23 departamentos existentes	'Idiomas'
		* director-departamento	VARCHAR(50)	Nombres de los 23 directores existentes	'Myriam Adriana Rodríguez López'
E9	ÁREA	# id-área	Integer	Entero entre 1 y 56 incluyendo extremos	42
		* nombre-área	VARCHAR(70)	Nombres de las 56 áreas existentes	'Medicina Estudiantil'
E10	FACULTAD	# id-facultad	Char(3)	Integridad por valor {AGR, JUR, ECO, ING, CIE}	'JUR'
		* nombre-facultad	VARCHAR(60)	Nombres de las 5 facultades existentes	'Ciencias Básicas y de la Educación'
		* decano-facultad	VARCHAR(50)	Nombres de los 5 decanos existentes	'Sandra Milena Piedrahita Buitrago'
E11	PROGRAMA	# id-programa	Char(10)	{'ZOO', 'AFS', 'SGI', 'ALI', 'MEC', 'EGA', 'LCNEA', 'AGR', 'DER', 'IND', 'BAT', 'VET', 'LROT', 'CIV', 'CON'}	'SGI'
		* nombre-programa	VARCHAR(80)	Nombres de los 16 programas existentes	'Medicina Veterinaria'

E12	SEDE	# id-sede	Integer	Entero entre 1 y 4 incluyendo extremos	4
		* nombre-sede	Varchar(70)	Nombres de las 4 sedes existentes	'Dependencia Teusaquillo'
E13	EDIFICIO	# id-edificio	Integer	Entero entre 1 y 18 incluyendo extremos	15
		* nombre-edificio	Varchar(50)	Nombres de los 18 edificios existentes	'Bloque B'
E14	LOCACIÓN	# id-locación	Integer	Entero entre 1 y 56 incluyendo extremos para salones. Para las oficinas, enteros de la forma AA00N donde AA es el identificador del área al que pertenece la oficina y N es el número de la oficina, sea 1 o 2. Enteros entre 1 y 9 incluyendo extremos para los laboratorios	34, 24002, 3
		* nombre-locación	Varchar(100)	Para los salones, nombres de la forma 'Salón EN' donde E es la inicial del edificio y N es el número del salón en ese edificio. Para las oficinas, nombres de la forma 'Oficina EN' donde E es la inicial del edificio y N es el número de la oficina. Para los laboratorios, es el conjunto de nombres de los laboratorios existentes	'Salón H34', 'Oficina C25002', 'Fitoquímica'
E15	OFICINA	* puestos-oficina	Integer	Entero entre 1 y 10 incluyendo extremos	4
	SALÓN	* capacidad-salón	Integer	Para las salas de cómputo {15, 16, 17, 18}, Para los auditorios {100, 110, 120, 130}, Para los salones {20, 30, 40, 50, 60}	17, 110, 20
		o tipo-salón	Char(3)	Integridad por valor {AUD, TAL, COM}	'AUD'
E17	LABORATORIO	o acreditación-laboratorio	Varchar(20)	{'IEC-ISO 15189', 'IEC-ISO 17025', 'En Proceso'}	'IEC-ISO 15189'

Tabla 12 - Detalle de atributos para cada entidad según la metodología de diseño E-R indicando el dominio definido para cada uno

Observaciones:

- No todos los activos vienen indicando el material, color, tamaño o componentes adicionales en su factura.
- Cada componente se separa por una coma y un espacio
- Algunos funcionarios son contratados a término indefinido
- Un área tiene una o dos oficinas.

Cuando se dice algo de la forma '*nombre de facultades existentes*' se refiere a los conjuntos de datos presentados en las especificaciones del diseño para la entidad en cuestión donde se mostraron las tablas de dominios.

Mnemotécnicas para los valores cuyo dominio está definido por siglas:

ACTIVO.CATEGORIA_ACTIVO

Los activos pueden ser clasificados en cinco categorías según sea el área que los administra.

- (CAE) Centro de Ayudas Educativas: Es el área que se administra los televisores, video beams y otros dispositivos que se encuentran en los salones para apoyar las clases.
- (DTS) División de Tecnología y Sistemas: También llamada Soporte Técnico y Telecomunicaciones, es el área que administra los computadores y otros dispositivos de las salas de cómputo y de las oficinas.
- (LOG) Logística y Servicios Administrativos: Es el departamento que administra los equipos que se utilizan en eventos estudiantiles y administrativos, como por ejemplo, los equipos de sonido de los auditorios.
- (SGT) Servicios Generales y Transportes: Es el área que administra el mobiliario de la universidad. Esto es, las sillas de los salones y oficinas y demás locaciones de la universidad.
- (LAB) Secciones de Laboratorios: Las tres secciones de laboratorio administran todos los equipos usados en sus instalaciones.

AUXILIAR.TIPO_AUXILIAR

- (INV) Auxiliar de Inventarios
- (MAN) Auxiliar de Mantenimiento
- (LAB) Auxiliar de Laboratorios

VICERRECTORIA.ID_VICERRECTORIA

- (FOR) Vicerrectoría de Formación
- (INV) Vicerrectoría de Investigación
- (EXT) Vicerrectoría de Extensión y Proyección Social
- (ADM) Vicerrectoría de Administración y Financiera

FACULTAD.ID_FACULTAD

- (AGR) Facultad de Ciencias Agrarias
- (JUR) Facultad de Ciencias Jurídicas y Sociales
- (ECO) Facultad de Ciencias Económicas, Administrativas y Contables
- (ING) Facultad de Ingeniería
- (ADM) Facultad de Ciencias Básicas y de la Educación

PROGRAMA.ID_PROGRAMA

- (AFS) Administración Financiera y de Sistemas
- (AGR) Ingeniería Agroindustrial
- (ALI) Ingeniería de Alimentos
- (BAT) Especialización en Bienestar Animal y Etología
- (CIV) Ingeniería Civil
- (CON) Contaduría Pública
- (DER) Derecho
- (EGA) Especialización en Gestión de Agronegocios
- (IND) Ingeniería Industrial
- (LCNEA) Licenciatura en Ciencias Naturales y Educación Ambiental
- (LROT) Especialización en Legislación Rural y Ordenamiento Territorial
- (MEC) Ingeniería Mecatrónica
- (SGI) Especialización en Sistemas de Gestión Integrada HSEQ, Especialización en Seguridad Industrial, Higiene y Gestión Ambiental

- (VET) Medicina Veterinaria
- (ZOO) Zootecnia

SALON.TIPO_SALON

Los salones de clase convencionales no tienen una sigla para distinguirlos. Estos salones tienen una capacidad de entre 20 y 60 personas.

- (AUD) Auditorios: Son salones con capacidad de entre 100 y 130 personas.
- (TAL) Talleres: Son salones especializados para algunas asignaturas o actividades de la universidad como por ejemplo la carpintería o la música. Su capacidad está entre las 20 y 60 personas.
- (COM) Salas de Cómputo: Son salas dedicadas a impartir clases que requieren del uso de computadores. Su capacidad está entre las 15 y 18 personas según el número de computadores disponibles.

Construcción del modelo entidad-relación inicial o modelo de datos conceptual CDM en PowerDesigner

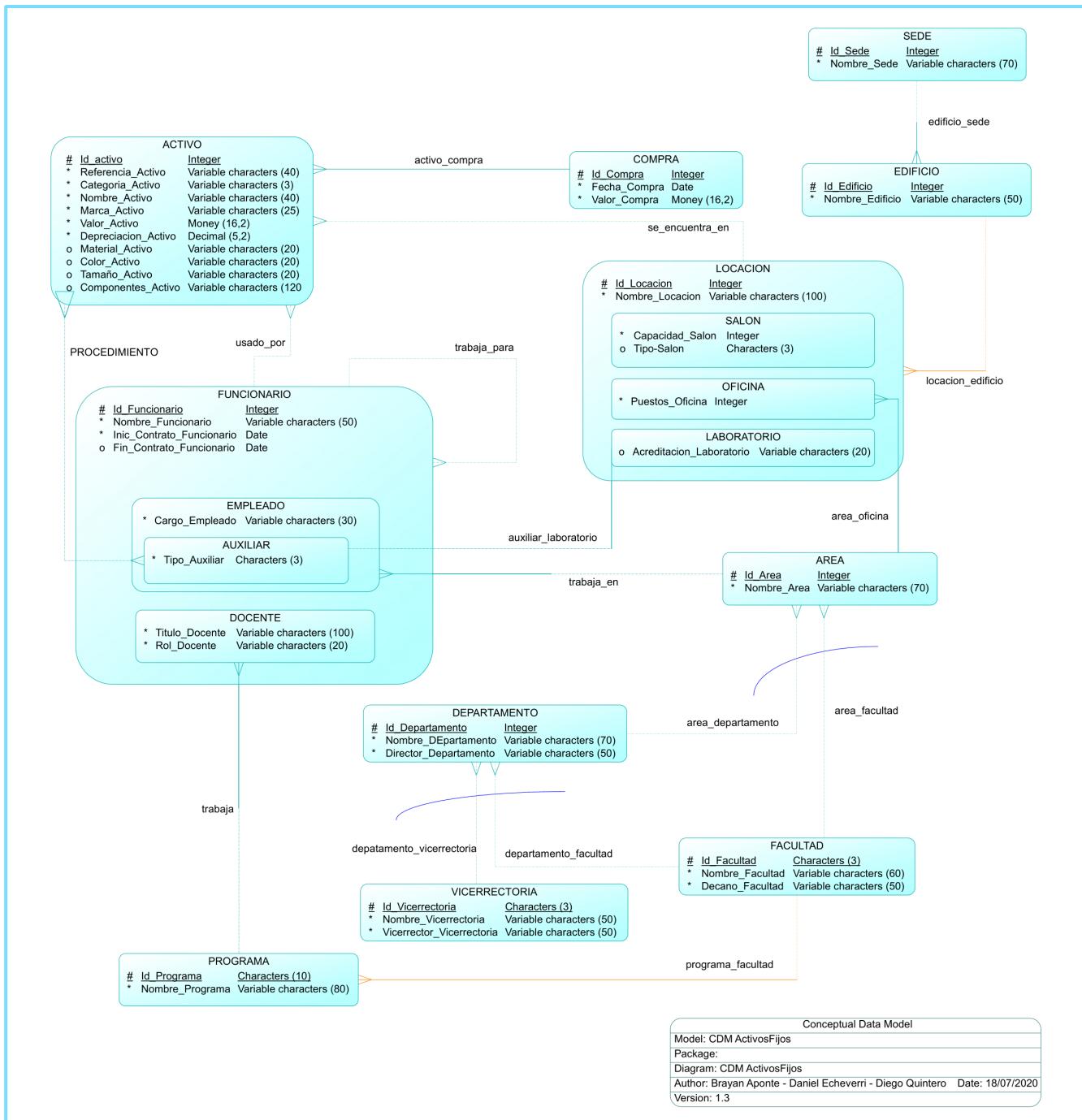


Figura 6 – Modelo de datos conceptual (CDM) o diagrama E-R inicial

Resolución de relaciones m-m para la generación de entidades intermedias

En el presente modelo se tiene una única relación de cardinalidad m-m, la cual es AUXILIAR-ACTIVO. Esta relación represente una serie de procedimientos llevados a cabo por los auxiliares de inventarios y de mantenimiento.

RELACIONES m-m		NOMBRE	CARDINALIDAD	PARTICIPACIÓN
E1 - E6	ACTIVO - AUXILIAR	PROCEDIMIENTO	m-m	> <

Se generará una entidad intermedia llamada PROCEDIMIENTO

ENTIDAD INTERMEDIA GENERADA		
E18	PROCEDIMIENTO	Llevará un registro de las entregas de puestos de trabajo, firmas de paz y salvos, revisiones y mantenimientos de activos fijos. El tipo de procedimiento a realizar dependerá del tipo de auxiliar

Esta nueva entidad tendrá una relación de cardinalidad m-1 con ACTIVO y AUXILIAR.

NUEVAS RELACIONES		NOMBRE	TIPO	CARDINALIDAD	PARTICIPACIÓN
E1 - E18	ACTIVO - PROCEDIMIENTO	ACTIVO-PROCEDIMIENTO	DEPENDENCIA	1-m	... —<
E6 - E18	AUXILIAR - PROCEDIMIENTO	AUXILIAR-PROCEDIMIENTO	DEPENDENCIA	1-m	... —<

Adicionalmente, se establecerá una nueva relación entre la entidad intermedia PROCEDIMIENTO y FUNCIONARIO. La cual busca generar una llave foránea en PROCEDIMIENTO que indique cuál es el funcionario al cual se le está haciendo una entrega del puesto de trabajo, uno de los procedimientos que tienen a cargo los auxiliares de inventario.

NUEVAS RELACIONES		NOMBRE	CARDINALIDAD	PARTICIPACIÓN
E3 - E18	FUNCIONARIO - PROCEDIMIENTO	ENTREGA	1-m <

A continuación se presenta la matriz de relaciones extendida, luego de este proceso:

MATRIZ DE RELACIONES																		
	E1	E2	E3	E4	E5	E6	E7	E8	E9	E10	E11	E12	E13	E14	E15	E16	E17	E18
E1		m-1	m-1			m-m								m-1				1-m
E2																		
E3			m-1															1-m
E4														m-1				
E5														m-1				
E6																	1-1	1-m
E7														m-1				
E8														1-m	m-1			
E9														m-1				1-m
E10														1-m				
E11																		
E12															1-m			
E13																1-m		
E14																		
E15																		
E16																		
E17																		
E18																		

Figura 5 – Matriz de relaciones final según la metodología de diseño E-R

Definición de atributos para las entidades intermedias generadas:

No se dejará la llave primaria compuesta generada de la relación m-m entre ACTIVO y AUXILIAR, sino que se asignará un atributo adicional que cumpla esta función:

ENTIDAD		ATRIBUTO	TIPO DE DATO	INTEGRIDAD POR VALOR
E18	PROCEDIMIENTO	# id-proce	Integer	
		* id-activo	Integer	
		* id-funcionario	Integer	
		* tipo-procedimiento	Char(1)	[E]ntrega, [P]az y salvo, [R]evisión, [M]antenimiento
		* fecha-procedimiento	Date	
		o estado-revision	Char(1)	[R]eemplazo, [M]antenimiento
		o observaciones-procedimiento	Varchar(150)	

Los conceptos de Entrega, Paz y salvo, Revisión, Mantenimiento y Reemplazo en el contexto del desarrollo fueron explicados en las especificaciones del diseño presentadas anteriormente en el documento.

Construcción del modelo entidad-relación final o modelo de datos lógico LDM en PowerDesigner

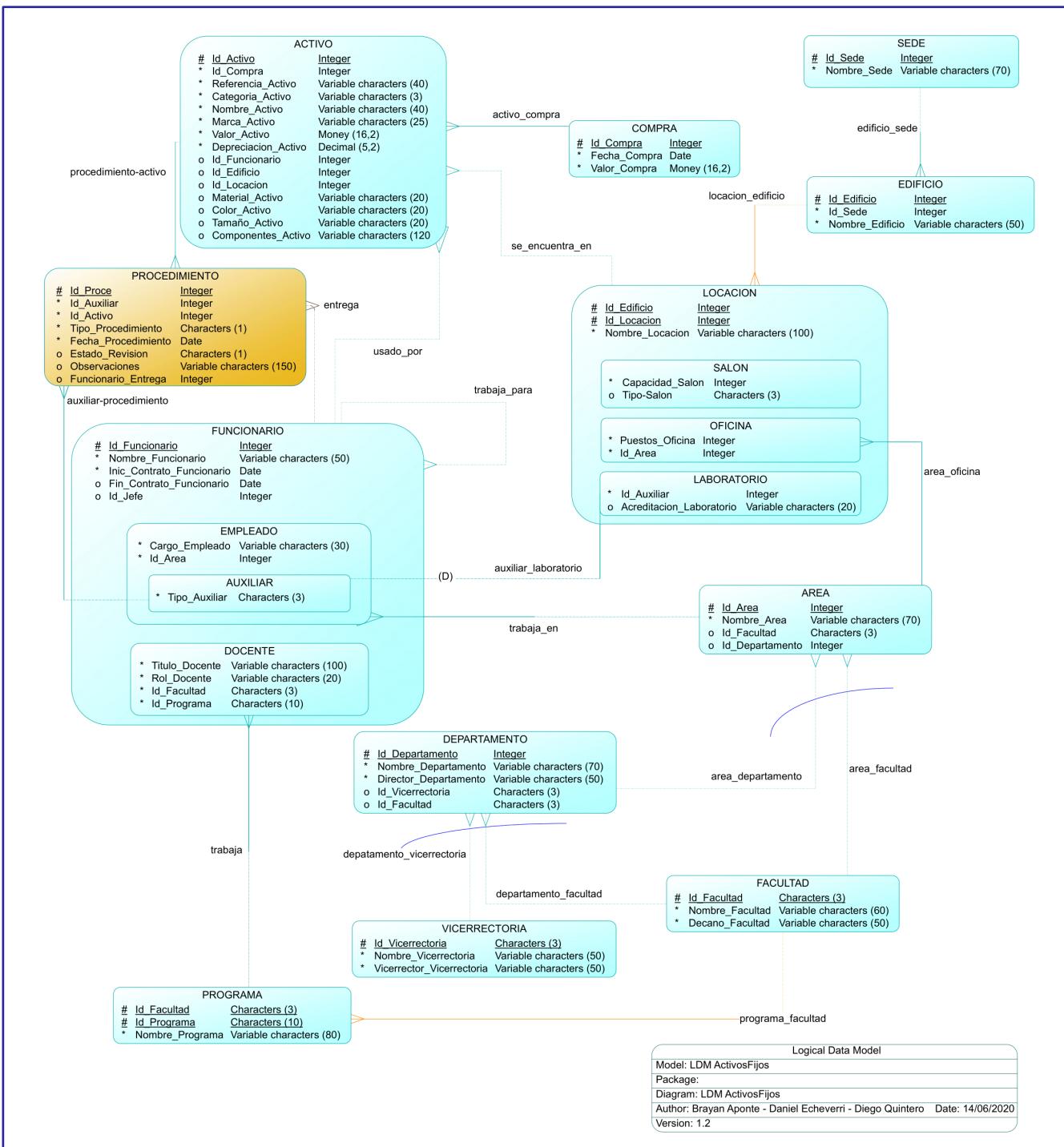


Figura 7 – Modelo de datos lógico (LDM) o diagrama E-R final

Para construir y validar este modelo de datos, fue necesario hacer las siguientes configuraciones:

Para los subtipos, es decir para AUXILIAR, EMPLEADO, DOCENTE, SALÓN, OFICINA y LABORATORIO, asegurarse que en los atributos aparezca marcado como llave primaria, el atributo destinado para ello y que pertenece al supertipo, es decir FUNCIONARIO y LOCACION.

A continuación se muestran dos ejemplos.

Entity Properties - AUXILIAR (AUXILIAR)									
	Name	Code	Data Typ	Length	Preci	M	P	D	
1	Tipo_Auxiliar	TIPO_AUXI	Characters (3)	3		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
→	Id_Funcionario	ID_FUNCIO	Integer			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
3	Nombre_Funcionario	NOMBRE_F	Variable char	50		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
4	Inic_Contrato_Func	INIC_CONT	Date			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
5	Fin_Contrato_Func	FIN CONTR	Date			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
6	Cargo_Emppleado	CARGO_EM	Variable char	30		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
7	Id_Jefe	ID_JEFE	Integer			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
8	Id_Area	ID_AREA	Integer			<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Figura 8 – Configuración para garantizar la integridad referencial en PowerDesigner cuando hay supertipos y subtipos para el modelo de datos lógico (LDM)

Entity Properties - SALON (SALON)									
	Name	Code	Data Typ	Length	Preci	M	P	D	
1	Id_Locacion	ID_LOCACI	Integer			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
2	Nombre_Locacion	NOMBRE_L	Variable char	100		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
→	Id_Edificio	ID_EDIFICI	Integer			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
4	Capacidad_Salon	CAPACIDAD	Integer			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
5	Tipo_Salon	TIPO_SALO	Characters (3)	3		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Figura 9 – Configuración para garantizar la integridad referencial en PowerDesigner cuando hay supertipos y subtipos para el modelo de datos lógico (LDM)

Marcar estas casillas donde aplique, debe garantizar que se puedan ver las llaves foráneas donde corresponde. Por ejemplo, a continuación se muestra como la llave foránea (compuesta) de laboratorio se presenta en AUXILIAR y la llave foránea del auxiliar se presenta en LABORATORIO.

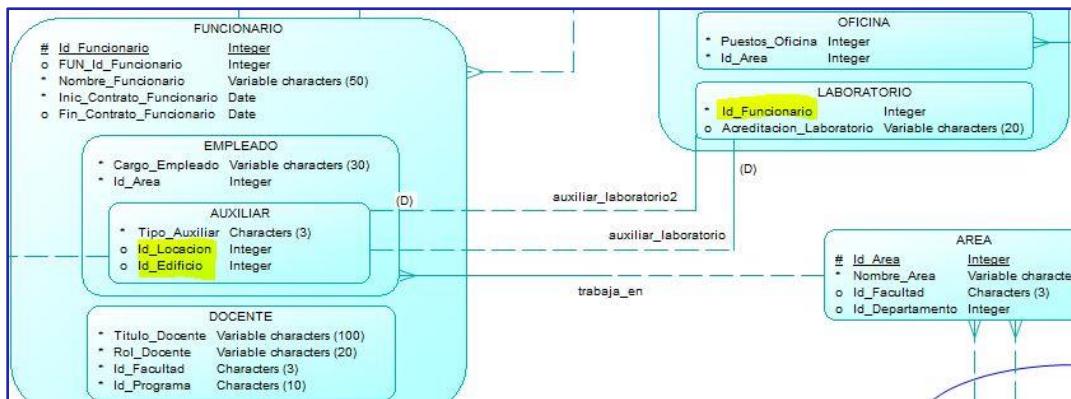


Figura 10 – Integridad referencial configurada apropiadamente en presencia de supertipos y subtipos para el modelo LDM

Particularmente con esta relación 1-1 entre AUXILIAR y LABORATORIO, se debe eliminar uno de los enlaces entre las dos entidades, para dejar que la llave foránea esté en solo una de las entidades.

Para el presente modelo, se eliminó el enlace *auxiliar_laboratorio* dejando así el enlace *auxiliar_laboratorio2*, al cual posteriormente se le modificó el nombre a *auxiliar_laboratorio* para guardar la notación. De esta manera se tiene la llave foránea en LABORATORIO únicamente, es decir, el extremo opcional de la relación, pues un AUXILIAR no necesariamente debe tener asignado un LABORATORIO, pero un LABORATORIO si debe tener asignado un AUXILIAR obligatoriamente.

Otras modificaciones realizadas corresponden a los nombres de los atributos generados como llaves foráneas en varias entidades. A continuación se mencionan:

- El atributo *id_funcionario* en LABORATORIO y PROCEDIMIENTO se cambió a *id_auxiliar*.
- El atributo *FUN_id_funcionario* en FUNCIONARIO que surgió de la relación recursiva se cambió a *id_jefe*

También se agregaron los atributos definidos para la relación intermedia junto con el enlace de la relación PROCEDIMIENTO-FUNCIONARIO, lo cual generó una llave foránea en PROCEDIMIENTO llamada *id_funcionario*, la cual se cambió a *funcionario_entrega*.

Todas estas modificaciones resultan en lo siguiente:

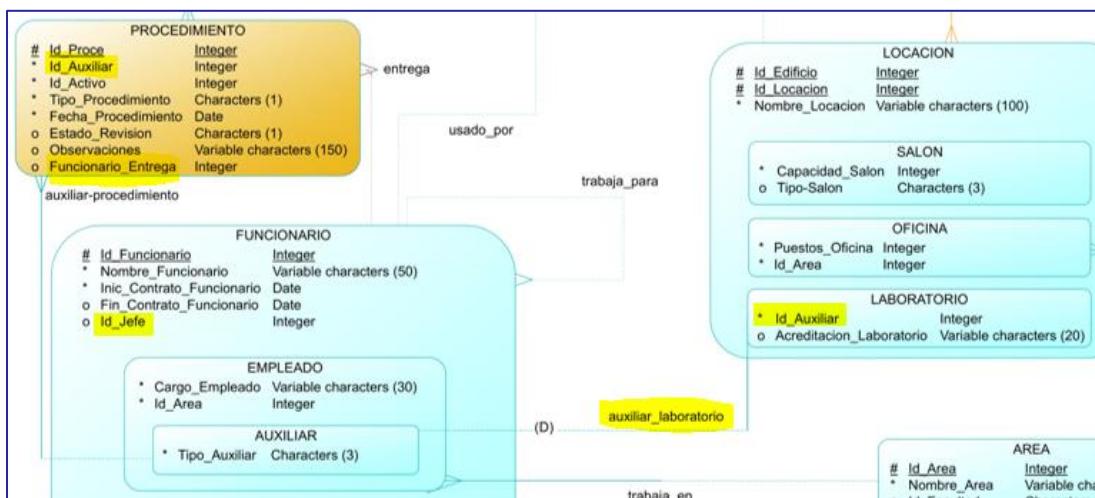


Figura 11 – Modificaciones finales al modelo LDM

Generación del modelo estructura-dato o modelo de datos físico PDM en PowerDesigner

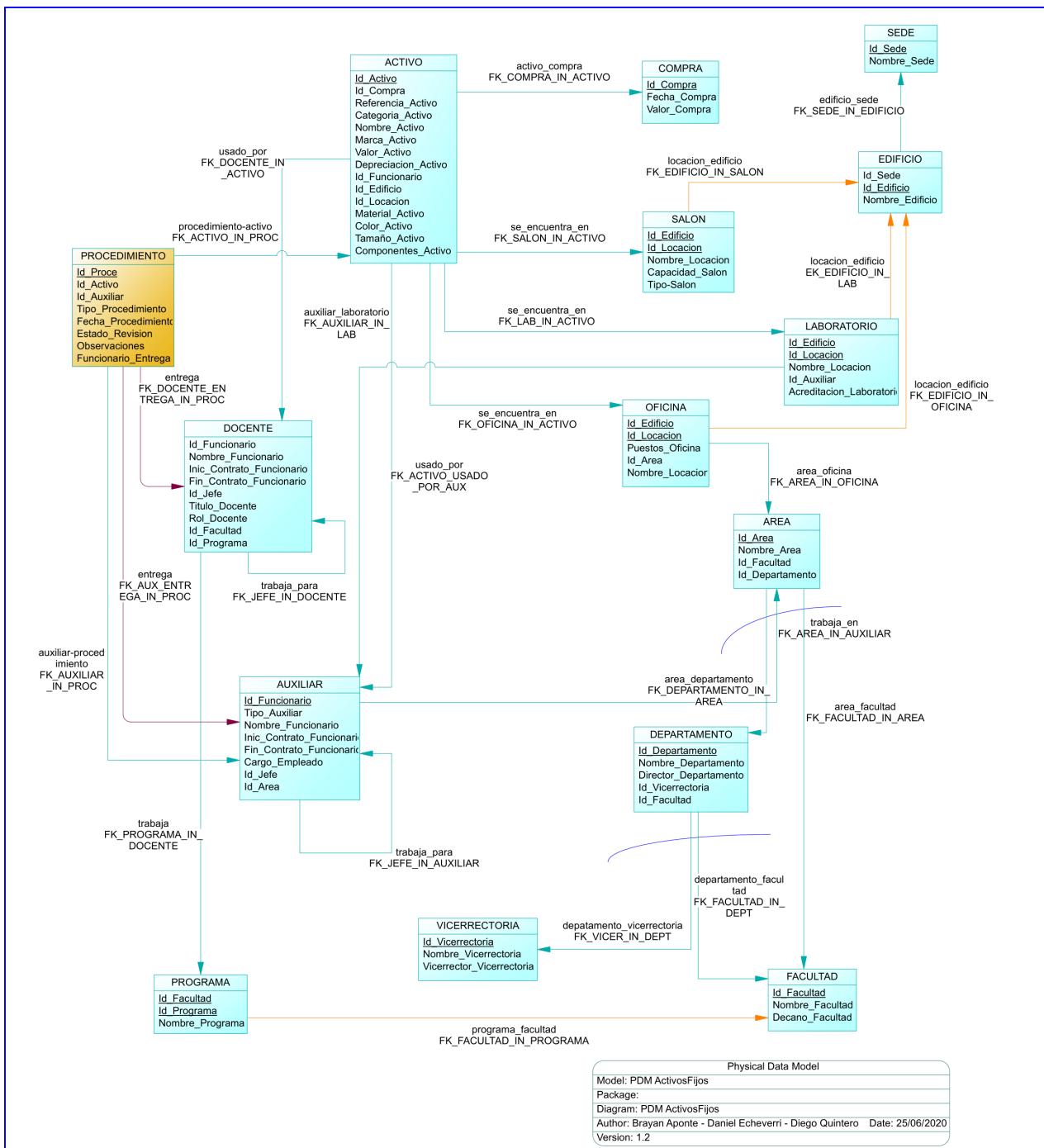


Figura 11 – Modelo de datos físico (PDM) o diagrama E-D

Generación del código de creación SQL inicial de la base de datos con PowerDesigner

A continuación se presenta el código de creación SQL de la base de datos según es generado por PowerDesigner. La versión usada para la generación del código fue ORACLE 11g.

Todo código SQL en el documento se presenta con el presente estilo.

```

/*=====
/* DBMS name:      ORACLE Version 11g          */
/* Created on:     25/06/2020 12:34:49 p. m.       */
/*=====

alter table ACTIVO
    drop constraint FK_COMPRA_IN_ACTIVO;

alter table ACTIVO
    drop constraint FK_OFICINA_IN_ACTIVO;

alter table ACTIVO
    drop constraint FK_LAB_IN_ACTIVO;

alter table ACTIVO
    drop constraint FK_SALON_IN_ACTIVO;

alter table ACTIVO
    drop constraint FK_ACTIVO_USADO_POR_AUX;

alter table ACTIVO
    drop constraint FK_DOCENTE_IN_ACTIVO;

alter table AREA
    drop constraint FK_DEPARTAMENTO_IN_AREA;

alter table AREA
    drop constraint FK_FACULTAD_IN_AREA;

alter table AUXILIAR
    drop constraint FK_AREA_IN_AUXILIAR;

alter table AUXILIAR
    drop constraint FK_JEFE_IN_AUXILIAR;

alter table DEPARTAMENTO
    drop constraint FK_FACULTAD_IN_DEPT;

alter table DEPARTAMENTO
    drop constraint FK_VICER_IN_DEPT;

alter table DOCENTE
    drop constraint FK_PROGRAMA_IN_DOCENTE;

alter table DOCENTE
    drop constraint FK_JEFE_IN_DOCENTE;

alter table EDIFICIO
    drop constraint FK_SEDE_IN_EDIFICIO;

alter table LABORATORIO
    drop constraint FK_AUXILIAR_IN_LAB;

alter table LABORATORIO
    drop constraint EK_EDIFICIO_IN_LAB;

alter table OFICINA

```

```
drop constraint FK_AREA_IN_OFICINA;
alter table OFICINA
  drop constraint FK_EDIFICIO_IN_OFICINA;
alter table PROCEDIMIENTO
  drop constraint FK_AUX_ENTREGA_IN_PROC;
alter table PROCEDIMIENTO
  drop constraint FK_DOCENTE_ENTREGA_IN_PROC;
alter table PROCEDIMIENTO
  drop constraint FK_AUXILIAR_IN_PROC;
alter table PROCEDIMIENTO
  drop constraint FK_ACTIVO_IN_PROC;
alter table PROGRAMA
  drop constraint FK_FACULTAD_IN_PROGRAMA;
alter table SALON
  drop constraint FK_EDIFICIO_IN_SALON;

drop index SE_ENCUENTRA_EN2_FK;
drop index USADO_POR2_FK;
drop index ACTIVO_COMPRA_FK;
drop table ACTIVO cascade constraints;
drop index AREA_DEPARTAMENTO_FK;
drop index AREA_FACULTAD_FK;
drop table AREA cascade constraints;
drop index TRABAJA_EN_FK;
drop table AUXILIAR cascade constraints;
drop table COMPRA cascade constraints;
drop index DEPARTAMENTO_FACULTAD_FK;
drop index DEPATAMENTO_VICERRECTORIA_FK;
drop table DEPARTAMENTO cascade constraints;
drop index TRABAJA_FK;
drop table DOCENTE cascade constraints;
drop index EDIFICIO_SEDE_FK;
drop table EDIFICIO cascade constraints;
drop table FACULTAD cascade constraints;
drop index LOCACION_EDIFICIO2_FK;
drop index AUXILIAR_LABORATORIO_FK;
drop table LABORATORIO cascade constraints;
drop index LOCACION_EDIFICIO_FK;
drop index AREA_OFICINA_FK;
drop table OFICINA cascade constraints;
drop index ENTREGA2_FK;
drop index PROCEDIMIENTO_ACTIVO_FK;
```

```

drop index PROCEDIMIENTO2_FK;
drop table PROCEDIMIENTO cascade constraints;
drop index PROGRAMA_FACULTAD_FK;
drop table PROGRAMA cascade constraints;
drop index LOCACION_EDIFICIO3_FK;
drop table SALON cascade constraints;
drop table SEDE cascade constraints;
drop table VICERRECTORIA cascade constraints;
/*=====
/* Table: ACTIVO
/*=====*/
create table ACTIVO
(
    ID_ACTIVO          INTEGER          not null,
    ID_COMPRA          INTEGER          not null,
    REFERENCIA_ACTIVO  VARCHAR2(40)     not null,
    CATEGORIA_ACTIVO   VARCHAR2(3)      not null
        constraint CKC_CATEGORIA_ACTIVO_ACTIVO check (CATEGORIA_ACTIVO in ('CAE','DTS','LAB','LOG','SGT')),
    NOMBRE_ACTIVO      VARCHAR2(40)     not null,
    MARCA_ACTIVO       VARCHAR2(25)     not null,
    VALOR_ACTIVO       NUMBER(16,2)     not null,
    DEPRECIACION_ACTIVO NUMBER(5,2)     not null,
    ID_FUNCIONARIO     INTEGER,
    ID_EDIFICIO        INTEGER,
    ID_LOCACION        INTEGER,
    MATERIAL_ACTIVO   VARCHAR2(20),
    COLOR_ACTIVO       VARCHAR2(20),
    TAMANO_ACTIVO      VARCHAR2(20),
    COMPONENTES_ACTIVO VARCHAR2(120),
    constraint PK_ACTIVO primary key (ID_ACTIVO)
);
/*=====
/* Index: ACTIVO_COMPRA_FK
/*=====*/
create index ACTIVO_COMPRA_FK on ACTIVO (
    ID_COMPRA ASC
);
/*=====
/* Index: USADO_POR2_FK
/*=====*/
create index USADO_POR2_FK on ACTIVO (
    ID_FUNCIONARIO ASC

```

```

);

/*=====
/* Index: SE_ENCUENTRA_EN2_FK
 */

create index SE_ENCUENTRA_EN2_FK on ACTIVO (
    ID_LOCACION ASC,
    ID_EDIFICIO ASC
);

/*=====
/* Table: AREA
 */

create table AREA
(
    ID_AREA          INTEGER          not null,
    NOMBRE_AREA      VARCHAR2(70)     not null,
    ID_FACULTAD      CHAR(3),
    ID_DEPARTAMENTO INTEGER,
    constraint PK_AREA primary key (ID_AREA)
);

/*=====
/* Index: AREA_FACULTAD_FK
 */

create index AREA_FACULTAD_FK on AREA (
    ID_FACULTAD ASC
);

/*=====
/* Index: AREA_DEPARTAMENTO_FK
 */

create index AREA_DEPARTAMENTO_FK on AREA (
    ID_DEPARTAMENTO ASC
);

/*=====
/* Table: AUXILIAR
 */

create table AUXILIAR
(
    ID_FUNCIONARIO   INTEGER          not null,
    TIPO_AUXILIAR    CHAR(3)         not null
        constraint CKC_TIPO_AUXILIAR_AUXILIAR check (TIPO_AUXILIAR in ('INV','MAN','LAB')),
    NOMBRE_FUNCIONARIO VARCHAR2(50),
    INIC_CONTRATO_FUNCIONARIO DATE,
    FIN_CONTRATO_FUNCIONARIO DATE,
);

```

```

CARGO_EMPLEADO      VARCHAR2(30),
ID_JEFE            INTEGER,
ID_AREA             INTEGER          not null,
constraint PK_AUXILIAR primary key (ID_FUNCIONARIO)
);

/*=====
/* Index: TRABAJA_EN_FK
*/
/*=====

create index TRABAJA_EN_FK on AUXILIAR (
    ID_AREA ASC
);

/*=====
/* Table: COMPRA
*/
/*=====

create table COMPRA
(
    ID_COMPRA          INTEGER          not null,
    FECHA_COMPRA       DATE            not null,
    VALOR_COMPRA        NUMBER(16,2)    not null,
constraint PK_COMPRA primary key (ID_COMPRA)
);

/*=====
/* Table: DEPARTAMENTO
*/
/*=====

create table DEPARTAMENTO
(
    ID_DEPARTAMENTO   INTEGER          not null,
    NOMBRE_DEPARTAMENTO  VARCHAR2(70)    not null,
    DIRECTOR_DEPARTAMENTO VARCHAR2(50)    not null,
    ID_VICERRECTORIA   CHAR(3),
    ID_FACULTAD         CHAR(3),
constraint PK_DEPARTAMENTO primary key (ID_DEPARTAMENTO)
);

/*=====
/* Index: DEPATAMENTO_VICERRECTORIA_FK
*/
/*=====

create index DEPATAMENTO_VICERRECTORIA_FK on DEPARTAMENTO (
    ID_VICERRECTORIA ASC
);

/*=====
/* Index: DEPARTAMENTO_FACULTAD_FK
*/
/*=====
```

```

create index DEPARTAMENTO_FACULTAD_FK on DEPARTAMENTO (
    ID_FACULTAD ASC
);

/*=====
/* Table: DOCENTE
*/
=====*/
create table DOCENTE
(
    ID_FUNCIONARIO      INTEGER          not null,
    NOMBRE_FUNCIONARIO  VARCHAR2(50),
    INIC_CONTRATO_FUNCIONARIO DATE,
    FIN_CONTRATO_FUNCIONARIO DATE,
    ID_JEFE              INTEGER,
    TITULO_DOCENTE       VARCHAR2(100)     not null,
    ROL_DOCENTE          VARCHAR2(20)      not null,
    ID_FACULTAD          CHAR(3)          not null,
    ID_PROGRAMA          CHAR(10)         not null,
    constraint AK_IDENTIFIER_1_DOCENTE unique (ID_FUNCIONARIO)
);

/*=====
/* Index: TRABAJA_FK
*/
=====*/
create index TRABAJA_FK on DOCENTE (
    ID_FACULTAD ASC,
    ID_PROGRAMA ASC
);

/*=====
/* Table: EDIFICIO
*/
=====*/
create table EDIFICIO
(
    ID_SEDE              INTEGER          not null,
    ID_EDIFICIO          INTEGER          not null,
    NOMBRE_EDIFICIO      VARCHAR2(50)     not null,
    constraint PK_EDIFICIO primary key (ID_EDIFICIO)
);

/*=====
/* Index: EDIFICIO_SEDE_FK
*/
=====*/
create index EDIFICIO_SEDE_FK on EDIFICIO (
    ID_SEDE ASC
);

```

```

/*=====
/* Table: FACULTAD
*/
/*=====

create table FACULTAD
(
    ID_FACULTAD      CHAR(3)          not null
        constraint CKC_ID_FACULTAD_FACULTAD check (ID_FACULTAD in ('AGR','JUR','ECO','ING','CIE')),
    NOMBRE_FACULTAD   VARCHAR2(60)     not null,
    DECANO_FACULTAD   VARCHAR2(50)     not null,
    constraint PK_FACULTAD primary key (ID_FACULTAD)
);

/*=====
/* Table: LABORATORIO
*/
/*=====

create table LABORATORIO
(
    ID_EDIFICIO       INTEGER          not null,
    ID_LOCACION        INTEGER          not null,
    NOMBRE_LOCACION    VARCHAR2(100),
    ID_AUXILIAR        INTEGER          not null,
    ACREDITACION_LABORATORIO VARCHAR2(20),
    constraint PK_LABORATORIO primary key (ID_LOCACION, ID_EDIFICIO),
    constraint AK_IDENTIFIER_1_LABORATO unique (ID_LOCACION)
);

/*=====
/* Index: AUXILIAR_LABORATORIO_FK
*/
/*=====

create index AUXILIAR_LABORATORIO_FK on LABORATORIO (
    ID_AUXILIAR ASC
);

/*=====
/* Index: LOCACION_EDIFICIO2_FK
*/
/*=====

create index LOCACION_EDIFICIO2_FK on LABORATORIO (
    ID_EDIFICIO ASC
);

/*=====
/* Table: OFICINA
*/
/*=====

create table OFICINA
(
    ID_EDIFICIO       INTEGER          not null,

```

```

ID_LOCACION      INTEGER          not null,
PUESTOS_OFICINA  INTEGER          not null,
ID_AREA           INTEGER          not null,
NOMBRE_LOCACION   VARCHAR2(100),
constraint PK_OFICINA primary key (ID_EDIFICIO, ID_LOCACION),
constraint AK_IDENTIFIER_1_OFICINA unique (ID_LOCACION)
);
/*=====
/* Index: AREA_OFICINA_FK
*/
/*=====
create index AREA_OFICINA_FK on OFICINA (
    ID_AREA ASC
);
/*=====
/* Index: LOCACION_EDIFICIO_FK
*/
/*=====
create index LOCACION_EDIFICIO_FK on OFICINA (
    ID_EDIFICIO ASC
);
/*=====
/* Table: PROCEDIMIENTO
*/
/*=====
create table PROCEDIMIENTO
(
    ID_PROCE        INTEGER          not null,
    ID_ACTIVO       INTEGER          not null,
    ID_AUXILIAR     INTEGER          not null,
    TIPO_PROCEDIMIENTO CHAR(1)       not null
        constraint CKC_TIPO_PROCEDIMIENTO_PROCEDIM check (TIPO_PROCEDIMIENTO in ('E','P','R','M')),
    FECHA_PROCEDIMIENTO DATE          not null,
    ESTADO_REVISION   CHAR(1)
        constraint CKC_ESTADO_REVISION_PROCEDIM check (ESTADO_REVISION is null or (ESTADO_REVISION in ('R','M'))),
    OBSERVACIONES    VARCHAR2(150),
    FUNCIONARIO_ENTREGA INTEGER,
    constraint PK_PROCEDIMIENTO primary key (ID_PROCE)
);
/*=====
/* Index: PROCEDIMIENTO2_FK
*/
/*=====
create index PROCEDIMIENTO2_FK on PROCEDIMIENTO (
    ID_AUXILIAR ASC
);

```

```

/*=====
/* Index: PROCEDIMIENTO_ACTIVO_FK
 */
/*=====*/
create index PROCEDIMIENTO_ACTIVO_FK on PROCEDIMIENTO (
    ID_ACTIVO ASC
);
/*=====
/* Index: ENTREGA2_FK
 */
/*=====*/
create index ENTREGA2_FK on PROCEDIMIENTO (
    FUNCIONARIO_ENTREGA ASC
);
/*=====
/* Table: PROGRAMA
 */
/*=====*/
create table PROGRAMA
(
    ID_FACULTAD      CHAR(3)          not null,
    ID_PROGRAMA       CHAR(10)         not null,
    NOMBRE_PROGRAMA  VARCHAR2(80)     not null,
    constraint PK_PROGRAMA primary key (ID_FACULTAD, ID_PROGRAMA)
);
/*=====
/* Index: PROGRAMA_FACULTAD_FK
 */
/*=====*/
create index PROGRAMA_FACULTAD_FK on PROGRAMA (
    ID_FACULTAD ASC
);
/*=====
/* Table: SALON
 */
/*=====*/
create table SALON
(
    ID_EDIFICIO      INTEGER         not null,
    ID_LOCACION      INTEGER         not null,
    NOMBRE_LOCACION  VARCHAR2(100),
    CAPACIDAD_SALON  INTEGER         not null,
    TIPO_SALON        CHAR(3)

    constraint CKC_TIPO_SALON_SALON check (TIPO_SALON is null or (TIPO_SALON in ('AUD','TAL','COM'))),
    constraint PK_SALON primary key (ID_LOCACION, ID_EDIFICIO),
    constraint AK_IDENTIFIER_1_SALON unique (ID_LOCACION)
);

```

```

/*=====
/* Index: LOCACION_EDIFICIO3_FK                               */
/*=====*/
create index LOCACION_EDIFICIO3_FK on SALON (
    ID_EDIFICIO ASC
);

/*=====
/* Table: SEDE                                         */
/*=====*/
create table SEDE
(
    ID_SEDE          INTEGER          not null,
    NOMBRE_SEDE      VARCHAR2(70)     not null,
    constraint PK_SEDE primary key (ID_SEDE)
);

/*=====
/* Table: VICERRECTORIA                                */
/*=====*/
create table VICERRECTORIA
(
    ID_VICERRECTORIA CHAR(3)        not null
        constraint CKC_ID_VICERRECTORIA_VICERREC check (ID_VICERRECTORIA in ('FOR','INV','EXT','ADM')),
    NOMBRE_VICERRECTORIA VARCHAR2(50)      not null,
    VICERRECTOR_VICERRECTORIA VARCHAR2(50)      not null,
    constraint PK_VICERRECTORIA primary key (ID_VICERRECTORIA)
);

alter table ACTIVO
    add constraint FK_COMPRA_IN_ACTIVO foreign key (ID_COMPRA)
        references COMPRA (ID_COMPRA);

alter table ACTIVO
    add constraint FK_OFICINA_IN_ACTIVO foreign key (ID_EDIFICIO, ID_LOCACION)
        references OFICINA (ID_EDIFICIO, ID_LOCACION);

alter table ACTIVO
    add constraint FK_LAB_IN_ACTIVO foreign key (ID_LOCACION, ID_EDIFICIO)
        references LABORATORIO (ID_LOCACION, ID_EDIFICIO);

alter table ACTIVO
    add constraint FK_SALON_IN_ACTIVO foreign key (ID_LOCACION, ID_EDIFICIO)
        references SALON (ID_LOCACION, ID_EDIFICIO);

alter table ACTIVO
    add constraint FK_ACTIVO_USADO_POR_AUX foreign key (ID_FUNCIONARIO)
        references AUXILIAR (ID_FUNCIONARIO);

```

```

alter table ACTIVO
  add constraint FK_DOCENTE_IN_ACTIVO foreign key (ID_FUNCIONARIO)
    references DOCENTE (ID_FUNCIONARIO);

alter table AREA
  add constraint FK_DEPARTAMENTO_IN_AREA foreign key (ID_DEPARTAMENTO)
    references DEPARTAMENTO (ID_DEPARTAMENTO);

alter table AREA
  add constraint FK_FACULTAD_IN_AREA foreign key (ID_FACULTAD)
    references FACULTAD (ID_FACULTAD);

alter table AUXILIAR
  add constraint FK_AREA_IN_AUXILIAR foreign key (ID_AREA)
    references AREA (ID_AREA);

alter table AUXILIAR
  add constraint FK_JEFE_IN_AUXILIAR foreign key (ID_JEFE)
    references AUXILIAR (ID_FUNCIONARIO);

alter table DEPARTAMENTO
  add constraint FK_FACULTAD_IN_DEPT foreign key (ID_FACULTAD)
    references FACULTAD (ID_FACULTAD);

alter table DEPARTAMENTO
  add constraint FK_VICER_IN_DEPT foreign key (ID_VICERRECTORIA)
    references VICERRECTORIA (ID_VICERRECTORIA);

alter table DOCENTE
  add constraint FK_PROGRAMA_IN_DOCENTE foreign key (ID_FACULTAD, ID_PROGRAMA)
    references PROGRAMA (ID_FACULTAD, ID_PROGRAMA);

alter table DOCENTE
  add constraint FK_JEFE_IN_DOCENTE foreign key (ID_JEFE)
    references DOCENTE (ID_FUNCIONARIO);

alter table EDIFICIO
  add constraint FK_SEDE_IN_EDIFICIO foreign key (ID_SEDE)
    references SEDE (ID_SEDE);

alter table LABORATORIO
  add constraint FK_AUXILIAR_IN_LAB foreign key (ID_AUXILIAR)
    references AUXILIAR (ID_FUNCIONARIO);

alter table LABORATORIO
  add constraint EK_EDIFICIO_IN_LAB foreign key (ID_EDIFICIO)
    references EDIFICIO (ID_EDIFICIO);

alter table OFICINA
  add constraint FK_AREA_IN_OFICINA foreign key (ID_AREA)
    references AREA (ID_AREA);

alter table OFICINA
  add constraint FK_EDIFICIO_IN_OFICINA foreign key (ID_EDIFICIO)
    references EDIFICIO (ID_EDIFICIO);

```

```

alter table PROCEDIMIENTO
    add constraint FK_AUX_ENTREGA_IN_PROC foreign key (FUNCIONARIO_ENTREGA)
        references AUXILIAR (ID_FUNCIONARIO);

alter table PROCEDIMIENTO
    add constraint FK_DOCENTE_ENTREGA_IN_PROC foreign key (FUNCIONARIO_ENTREGA)
        references DOCENTE (ID_FUNCIONARIO);

alter table PROCEDIMIENTO
    add constraint FK_AUXILIAR_IN_PROC foreign key (ID_AUXILIAR)
        references AUXILIAR (ID_FUNCIONARIO);

alter table PROCEDIMIENTO
    add constraint FK_ACTIVO_IN_PROC foreign key (ID_ACTIVO)
        references ACTIVO (ID_ACTIVO);

alter table PROGRAMA
    add constraint FK_FACULTAD_IN_PROGRAMA foreign key (ID_FACULTAD)
        references FACULTAD (ID_FACULTAD);

alter table SALON
    add constraint FK_EDIFICIO_IN_SALON foreign key (ID_EDIFICIO)
        references EDIFICIO (ID_EDIFICIO);

```

Resolución de supertipos y subtipos:

En el presente modelo se tienen dos supertipos: FUNCIONARIO y LOCACIÓN. Las entidades EMPLEADO, AUXILIAR, DOCENTE, SALÓN, OFICINA y LABORATORIO son especializaciones de estos supertipos.

El método a utilizar será el de generar una tabla única para cada supertipo y sus subtipos.

El primer paso consiste en diseñar una estructura general para cada supertipo, que contenga todos los atributos de sí mismo y de sus subtipos.

Supertipo FUNCIONARIO

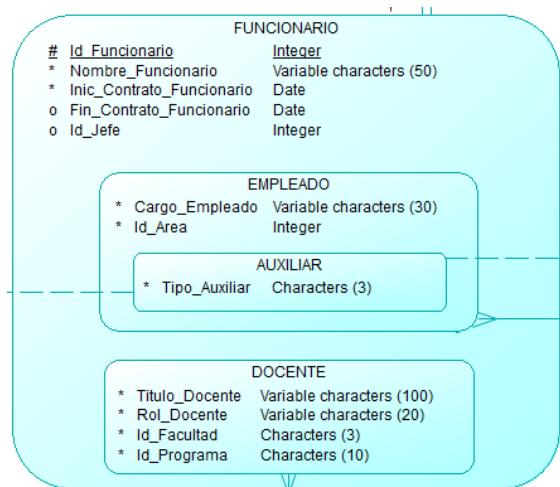


Figura 12 – Supertipo FUNCIONARIO según fue diseñado en PowerDesigner

Se diseña una tabla FUNCIONARIO que recoja todos los atributos de la imagen anterior junto con un atributo *tipo_funcionario*. Los atributos obligatorios de los subtipos pasarán a ser opcionales.

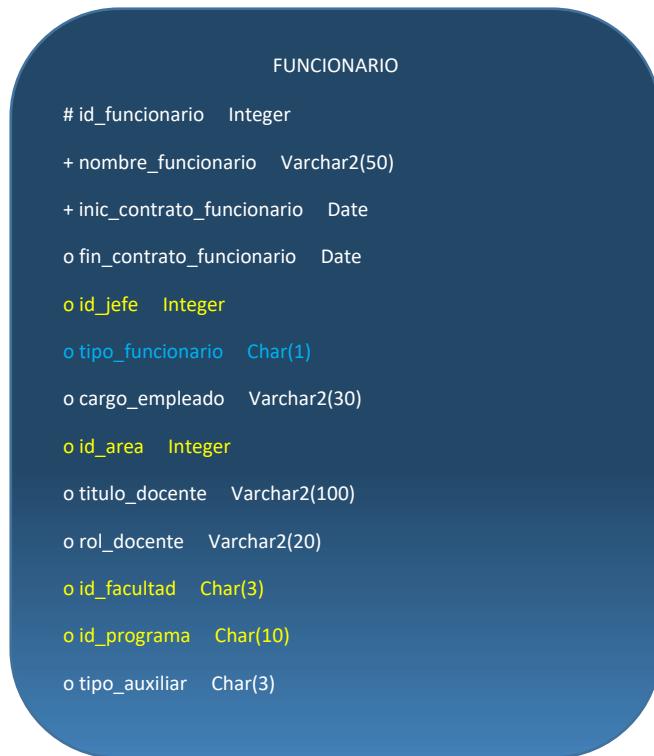


Figura 13 – Supertipo FUNCIONARIO para la resolución de supertipos y subtipos

La integridad por valor para el atributo *tipo_funcionario* será ‘E’ para empleado y ‘D’ para docente.

Los atributos en color amarillo representan llaves foráneas.

Supertipo LOCACION

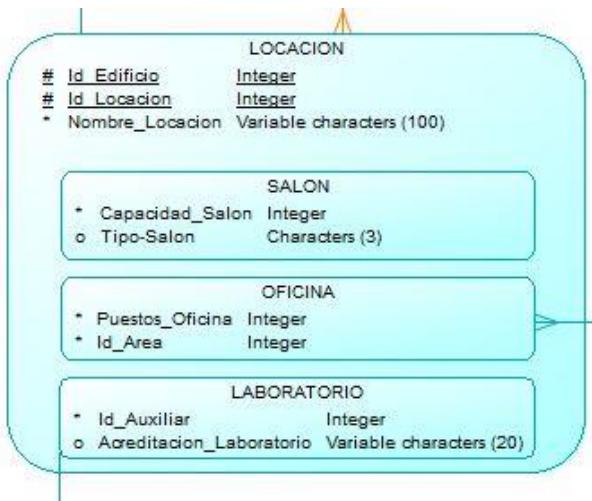


Figura 14 – Supertipo LOCACION según fue diseñado en PowerDesigner

Se diseña una tabla LOCACION que recoja todos los atributos anteriores y uno adicional llamado **tipo_locacion**, Los atributos obligatorios de los subtipos pasarán a ser opcionales.

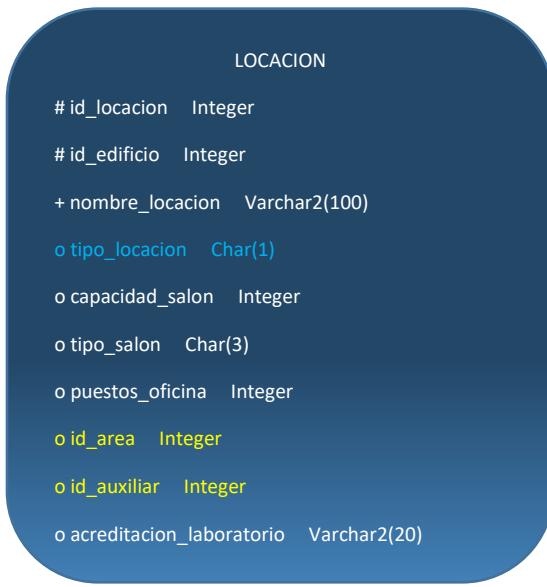


Figura 15 – Supertipo LOCACION para la resolución de supertipos y subtipos

La integridad por valor para el atributo **tipo_locacion** será ‘S’ para salón, ‘O’ para oficina y ‘L’ para laboratorio.

Los atributos en color amarillo representan llaves foráneas.

El segundo paso consiste en proceder a realizar las modificaciones en el código de creación generado como producto del diseño.

A continuación se presenta el SQL de creación completo. Se marcarán en color rojo las líneas a eliminar y en color verde las líneas a agregar o modificar.

```

/*
=====
/* DBMS name:      ORACLE Version 11g
   * /
/* Created on:    25/06/2020 12:34:49 p. m.
   * /

```

```
/*=====
alter table ACTIVO
    drop constraint FK_COMPRA_IN_ACTIVO;
alter table ACTIVO
    drop constraint FK_LOCACION_IN_ACTIVO;
alter table ACTIVO
    drop constraint FK_LAB_IN_ACTIVO;
alter table ACTIVO
    drop constraint FK_SALON_IN_ACTIVO;
alter table ACTIVO
    drop constraint FK_ACTIVOUSADO_POR_FUN;
alter table ACTIVO
    drop constraint FK_DOCENTE_IN_ACTIVO;
alter table AREA
    drop constraint FK_DEPARTAMENTO_IN_AREA;
alter table AREA
    drop constraint FK_FACULTAD_IN_AREA;
alter table FUNCIONARIO
    drop constraint FK_AREA_IN_FUNCIONARIO;
alter table FUNCIONARIO
    drop constraint FK_JEFE_IN_FUNCIONARIO;
alter table DEPARTAMENTO
    drop constraint FK_FACULTAD_IN_DEPT;
alter table DEPARTAMENTO
    drop constraint FK_VICER_IN_DEPT;
alter table FUNCIONARIO
    drop constraint FK_PROGRAMA_IN_FUNCIONARIO;
alter table DOCENTE
    drop constraint FK_JEFE_IN_DOCENTE;
alter table EDIFICIO
    drop constraint FK_SEDE_IN_EDIFICIO;
alter table LOCACION
    drop constraint FK_AUXILIAR_IN_LAB;
alter table LABORATORIO
    drop constraint EK_EDIFICIO_IN_LAB;
alter table LOCACION
    drop constraint FK_AREA_IN_OFICINA;
alter table LOCACION
    drop constraint FK_EDIFICIO_IN_LOCACION;
alter table PROCEDIMIENTO
    drop constraint FK_FUN_ENTREGA_IN_PROC;
alter table PROCEDIMIENTO
```

```
drop constraint FK_DOCENTE_ENTREGA_IN_PROC;

alter table PROCEDIMIENTO
    drop constraint FK_FUNCIONARIO_IN_PROC;

alter table PROCEDIMIENTO
    drop constraint FK_ACTIVO_IN_PROC;

alter table PROGRAMA
    drop constraint FK_FACULTAD_IN_PROGRAMA;

alter table SALON
    drop constraint FK_EDIFICIO_IN_SALON;

drop index SE_ENCUENTRA_EN2_FK;
drop index USADO_POR2_FK;
drop index ACTIVO_COMPRA_FK;
drop table ACTIVO cascade constraints;
drop index AREA_DEPARTAMENTO_FK;
drop index AREA_FACULTAD_FK;
drop table AREA cascade constraints;
drop index TRABAJA_EN_FK;
drop table FUNCIONARIO cascade constraints;
drop table COMPRA cascade constraints;
drop index DEPARTAMENTO_FACULTAD_FK;
drop index DEPATAMENTO_VICERRECTORIA_FK;
drop table DEPARTAMENTO cascade constraints;
drop index TRABAJA_FK;
drop table DOCENTE cascade constraints;
drop index EDIFICIO_SEDE_FK;
drop table EDIFICIO cascade constraints;
drop table FACULTAD cascade constraints;
drop index LOCACION_EDIFICIO2_FK;
drop index AUXILIAR_LABORATORIO_FK;
drop table LABORATORIO cascade constraints;
drop index LOCACION_EDIFICIO_FK;
drop index AREA_OFICINA_FK;
drop table LOCACION cascade constraints;
drop index ENTREGA2_FK;
drop index PROCEDIMIENTO_ACTIVO_FK;
drop index PROCEDIMIENTO2_FK;
drop table PROCEDIMIENTO cascade constraints;
drop index PROGRAMA_FACULTAD_FK;
drop table PROGRAMA cascade constraints;
drop index LOCACION_EDIFICIO3_FK;
drop table SALON cascade constraints;
```

```

drop table SEDE cascade constraints;
drop table VICERRECTORIA cascade constraints;

/*
 * Table: ACTIVO
 */
/*
create table ACTIVO

(
    ID_ACTIVO          INTEGER          not null,
    ID_COMPRA          INTEGER          not null,
    REFERENCIA_ACTIVO  VARCHAR2(40)     not null,
    CATEGORIA_ACTIVO   VARCHAR2(3)      not null
        constraint CKC_CATEGORIA_ACTIVO_ACTIVO check (CATEGORIA_ACTIVO in ('CAE','DTS','LAB','LOG','SGT')),
    NOMBRE_ACTIVO       VARCHAR2(40)     not null,
    MARCA_ACTIVO        VARCHAR2(25)     not null,
    VALOR_ACTIVO        NUMBER(16,2)     not null,
    DEPRECIACION_ACTIVO NUMBER(5,2)      not null,
    ID_FUNCIONARIO      INTEGER,
    ID_EDIFICIO         INTEGER,
    ID_LOCACION         INTEGER,
    MATERIAL_ACTIVO    VARCHAR2(20),
    COLOR_ACTIVO        VARCHAR2(20),
    TAMANO_ACTIVO       VARCHAR2(20),
    COMPONENTES_ACTIVO VARCHAR2(120),
    constraint PK_ACTIVO primary key (ID_ACTIVO)
);

/*
 * Index: ACTIVO_COMPRA_FK
 */
/*
create index ACTIVO_COMPRA_FK on ACTIVO (
    ID_COMPRA ASC
);
/*
 * Index: USADO_POR2_FK
 */
/*
create index USADO_POR2_FK on ACTIVO (
    ID_FUNCIONARIO ASC
);
/*
 * Index: SE_ENCUENTRA_EN2_FK
 */
/*
create index SE_ENCUENTRA_EN2_FK on ACTIVO (

```

```

    ID_LOCACION ASC,
    ID_EDIFICIO ASC
);

/*=====
/* Table: AREA
*/
/*=====

create table AREA
(
    ID_AREA          INTEGER          not null,
    NOMBRE_AREA      VARCHAR2(70)     not null,
    ID_FACULTAD     CHAR(3),
    ID_DEPARTAMENTO INTEGER,
    constraint PK_AREA primary key (ID_AREA)
);

/*=====
/* Index: AREA_FACULTAD_FK
*/
/*=====

create index AREA_FACULTAD_FK on AREA (
    ID_FACULTAD ASC
);

/*=====
/* Index: AREA_DEPARTAMENTO_FK
*/
/*=====

create index AREA_DEPARTAMENTO_FK on AREA (
    ID_DEPARTAMENTO ASC
);

/*=====
/* Table: FUNCIONARIO
*/
/*=====

create table FUNCIONARIO
(
    ID_FUNCIONARIO   INTEGER          not null,
    NOMBRE_FUNCIONARIO VARCHAR2(50)    not null,
    INIC_CONTRATO_FUNCIONARIO DATE      not null,
    FIN_CONTRATO_FUNCIONARIO DATE      ,
    ID_JEFE          INTEGER          ,
    TIPO_FUNCIONARIO CHAR(1)          not null
    constraint CKC_TIPO_FUN check (TIPO_FUNCIONARIO in ('E', 'D')),
    CARGO_EMPLEADO    VARCHAR2(30)      ,
    ID_AREA           INTEGER          ,
    TITULO_DOCENTE    VARCHAR2(100)     ,
    ROL_DOCENTE       VARCHAR2(20)      ,
);

```

```

    ID_FACULTAD      CHAR(3)          ,
    ID_PROGRAMA       CHAR(10)         ,
    TIPO_AUXILIAR    CHAR(3)          ,

    constraint CKC_TIPO_AUXILIAR_FUN check (TIPO_AUXILIAR in ('INV','MAN','LAB')) ,
    constraint PK_FUNCIONARIO primary key (ID_FUNCIONARIO)

);

/*=====
/* Index: TRABAJA_EN_FK
/*=====

create index TRABAJA_EN_FK on FUNCIONARIO (
    ID_AREA ASC
);

/*=====

/* Table: COMPRA
/*=====

create table COMPRA
(
    ID_COMPRA        INTEGER          not null,
    FECHA_COMPRA    DATE             not null,
    VALOR_COMPRA    NUMBER(16,2)     not null,
    constraint PK_COMPRA primary key (ID_COMPRA)
);

/*=====

/* Table: DEPARTAMENTO
/*=====

create table DEPARTAMENTO
(
    ID_DEPARTAMENTO   INTEGER          not null,
    NOMBRE_DEPARTAMENTO  VARCHAR2(70)    not null,
    DIRECTOR_DEPARTAMENTO VARCHAR2(50)    not null,
    ID_VICERRECTORIA   CHAR(3),
    ID_FACULTAD        CHAR(3),
    constraint PK_DEPARTAMENTO primary key (ID_DEPARTAMENTO)
);

/*=====

/* Index: DEPARTAMENTO_VICERRECTORIA_FK
/*=====

create index DEPARTAMENTO_VICERRECTORIA_FK on DEPARTAMENTO (
    ID_VICERRECTORIA ASC
);

/*=====

/* Index: DEPARTAMENTO_FACULTAD_FK
*/

```

```

/*=====
create index DEPARTAMENTO_FACULTAD_FK on DEPARTAMENTO (
    ID_FACULTAD ASC
);

/*=====
/* Table: DOCENTE
*/
/*=====

create table DOCENTE
(
    ID_FUNCIONARIO      INTEGER          not null,
    NOMBRE_FUNCIONARIO   VARCHAR2(50),
    INIC_CONTRATO_FUNCIONARIO DATE,
    FIN_CONTRATO_FUNCIONARIO DATE,
    ID_JEFE              INTEGER,
    TITULO_DOCENTE       VARCHAR2(100)     not null,
    ROL_DOCENTE          VARCHAR2(20)      not null,
    ID_FACULTAD          CHAR(3)          not null,
    ID_PROGRAMA          CHAR(10)         not null,
    constraint AK_IDENTIFIER_1_DOCENTE unique (ID_FUNCIONARIO)
);

/*=====
/* Index: TRABAJA_FK
*/
/*=====

create index TRABAJA_FK on DOCENTE (
    ID_FACULTAD ASC,
    ID_PROGRAMA ASC
);

/*=====
/* Table: EDIFICIO
*/
/*=====

create table EDIFICIO
(
    ID_SEDE              INTEGER          not null,
    ID_EDIFICIO          INTEGER          not null,
    NOMBRE_EDIFICIO      VARCHAR2(50)     not null,
    constraint PK_EDIFICIO primary key (ID_EDIFICIO)
);

/*=====
/* Index: EDIFICIO_SEDE_FK
*/
/*=====

create index EDIFICIO_SEDE_FK on EDIFICIO (
    ID_SEDE ASC
);

```

```

);

/*=====
/* Table: FACULTAD
*/
=====*/

create table FACULTAD
(
    ID_FACULTAD      CHAR(3)          not null
        constraint CKC_ID_FACULTAD_FACULTAD check (ID_FACULTAD in ('AGR','JUR','ECO','ING','CIE')),

    NOMBRE_FACULTAD   VARCHAR2(60)       not null,
    DECANO_FACULTAD   VARCHAR2(50)       not null,
    constraint PK_FACULTAD primary key (ID_FACULTAD)
);

/*=====*/
/* Table: LABORATORIO
*/
=====*/

create table LABORATORIO
(
    ID_EDIFICIO      INTEGER           not null,
    ID_LOCACION       INTEGER           not null,
    NOMBRE_LOCACION   VARCHAR2(100),
    ID_AUXILIAR       INTEGER           not null,
    ACREDITACION_LABORATORIO VARCHAR2(20),
    constraint PK_LABORATORIO primary key (ID_LOCACION, ID_EDIFICIO),
    constraint AK_IDENTIFIER_1_LABORATO unique (ID_LOCACION)
);

/*=====*/
/* Index: LOCACION_EDIFICIO2_FK
*/
=====*/

create index LOCACION_EDIFICIO2_FK on LABORATORIO (
    ID_EDIFICIO ASC
);

/*=====*/
/* Table: LOCACION
*/
=====*/

create table LOCACION
(
    ID_EDIFICIO      INTEGER           not null,
    ID_LOCACION       INTEGER           not null,
    NOMBRE_LOCACION   VARCHAR2(100)      not null,
    TIPO_LOCACION     CHAR(1)            not null
        constraint CKC_TIPO_LOCACION check (TIPO_LOCACION in ('S','O','L')),

    CAPACIDAD_SALON   INTEGER           ,
    constraint PK_LOCACION primary key (ID_EDIFICIO, ID_LOCACION)
);

```

```

TIPO_SALON           CHAR(3)

constraint CKC_TIPO_SALON check (TIPO_SALON is null or (TIPO_SALON in ('AUD','TAL','COM'))),

PUESTOS_OFICINA     INTEGER          ,
ID_AREA              INTEGER          ,
ID_AUXILIAR          INTEGER          ,
ACREDITACION_LAB    VARCHAR2(20)      ,

constraint PK_LOCACION primary key (ID_EDIFICIO, ID_LOCACION)_

constraint AK_IDENTIFIER_1_OFICINA unique (ID_LOCACION)

);

/*=====
/* Index: AUXILIAR_LABORATORIO_FK
/*=====

create index AUXILIAR_LABORATORIO_FK on LOCACION (
    ID_AUXILIAR ASC
);

/*=====
/* Index: AREA_OFICINA_FK
/*=====

create index AREA_OFICINA_FK on LOCACION (
    ID_AREA ASC
);

/*=====
/* Index: LOCACION_EDIFICIO_FK
/*=====

create index LOCACION_EDIFICIO_FK on LOCACION (
    ID_EDIFICIO ASC
);

/*=====
/* Table: PROCEDIMIENTO
/*=====

create table PROCEDIMIENTO

(
    ID_PROCE            INTEGER          not null,
    ID_ACTIVO           INTEGER          not null,
    ID_AUXILIAR         INTEGER          not null,
    TIPO_PROCEDIMIENTO CHAR(1)         not null

constraint CKC_TIPO_PROCEDIMIENTO_PROCEDIM check (TIPO_PROCEDIMIENTO in ('E','P','R','M')),

FECHA_PROCEDIMIENTO DATE             not null,
ESTADO_REVISION       CHAR(1)

constraint CKC_ESTADO_REVISION_PROCEDIM check (ESTADO_REVISION is null or (ESTADO_REVISION in ('R','M'))),
OBSERVACIONES        VARCHAR2(150),
FUNCIONARIO_ENTREGA  INTEGER,

```

```

constraint PK_PROCEDIMIENTO primary key (ID_PROCE)
);

/*=====
 * Index: PROCEDIMIENTO2_FK
 */

create index PROCEDIMIENTO2_FK on PROCEDIMIENTO (
    ID_AUXILIAR ASC
);

/*=====
 * Index: PROCEDIMIENTO_ACTIVO_FK
 */

create index PROCEDIMIENTO_ACTIVO_FK on PROCEDIMIENTO (
    ID_ACTIVO ASC
);

/*=====
 * Index: ENTREGA2_FK
 */

create index ENTREGA2_FK on PROCEDIMIENTO (
    FUNCIONARIO_ENTREGA ASC
);

/*=====
 * Table: PROGRAMA
 */

create table PROGRAMA

(
    ID_FACULTAD      CHAR(3)          not null,
    ID_PROGRAMA       CHAR(10)         not null,
    NOMBRE_PROGRAMA  VARCHAR2(80)     not null,
    constraint PK_PROGRAMA primary key (ID_FACULTAD, ID_PROGRAMA)
);

/*=====
 * Index: PROGRAMA_FACULTAD_FK
 */

create index PROGRAMA_FACULTAD_FK on PROGRAMA (
    ID_FACULTAD ASC
);

/*=====
 * Table: SALON
 */

create table SALON

(
    ID_EDIFICIO      INTEGER         not null,

```

```

ID_LOCACION      INTEGER          not null,
NOMBRE_LOCACION  VARCHAR2(100),
CAPACIDAD_SALON  INTEGER          not null,
TIPO_SALON        CHAR(3)

constraint CKC_TIPO_SALON_SALON check (TIPO_SALON is null or (TIPO_SALON in ('AUD','TAL','COM'))),
constraint PK_SALON primary key (ID_LOCACION, ID_EDIFICIO),
constraint AK_IDENTIFIER_1_SALON unique (ID_LOCACION)

);

/*=====
* Index: LOCACION_EDIFICIO3_FK
*/
/*=====*/

create index LOCACION_EDIFICIO3_FK on SALON (
    ID_EDIFICIO ASC
);

/*=====
* Table: SEDE
*/
/*=====*/

create table SEDE

(
    ID_SEDE      INTEGER          not null,
    NOMBRE_SEDE   VARCHAR2(70)     not null,
    constraint PK_SEDE primary key (ID_SEDE)
);

/*=====
* Table: VICERRECTORIA
*/
/*=====*/

create table VICERRECTORIA

(
    ID_VICERRECTORIA  CHAR(3)      not null
        constraint CKC_ID_VICERRECTORIA_VICERREC check (ID_VICERRECTORIA in ('FOR','INV','EXT','ADM')),
    NOMBRE_VICERRECTORIA VARCHAR2(50)     not null,
    VICERRECTOR_VICERRECTORIA VARCHAR2(50)     not null,
    constraint PK_VICERRECTORIA primary key (ID_VICERRECTORIA)
);

alter table ACTIVO
    add constraint FK_COMPRA_IN_ACTIVO foreign key (ID_COMPRA)
        references COMPRA (ID_COMPRA);

alter table ACTIVO
    add constraint FK_LOCACION_IN_ACTIVO foreign key (ID_EDIFICIO, ID_LOCACION)
        references LOCACION (ID_EDIFICIO, ID_LOCACION);

alter table ACTIVO

```

```

add constraint FK_LAB_IN_ACTIVO foreign key (ID_LOCACION, ID_EDIFICIO)
    references LABORATORIO (ID_LOCACION, ID_EDIFICIO);

alter table ACTIVO
    add constraint FK_SALON_IN_ACTIVO foreign key (ID_LOCACION, ID_EDIFICIO)
        references SALON (ID_LOCACION, ID_EDIFICIO);

alter table ACTIVO
    add constraint FK_ACTIVO_USADO_POR_FUN foreign key (ID_FUNCIONARIO)
        references FUNCIONARIO (ID_FUNCIONARIO);

alter table ACTIVO
    add constraint FK_DOCENTE_IN_ACTIVO foreign key (ID_FUNCIONARIO)
        references DOCENTE (ID_FUNCIONARIO);

alter table AREA
    add constraint FK_DEPARTAMENTO_IN_AREA foreign key (ID_DEPARTAMENTO)
        references DEPARTAMENTO (ID_DEPARTAMENTO);

alter table AREA
    add constraint FK_FACULTAD_IN_AREA foreign key (ID_FACULTAD)
        references FACULTAD (ID_FACULTAD);

alter table FUNCIONARIO
    add constraint FK_AREA_IN_FUNCIONARIO foreign key (ID_AREA)
        references AREA (ID_AREA);

alter table FUNCIONARIO
    add constraint FK_JEFE_IN_FUNCIONARIO foreign key (ID_JEFE)
        references FUNCIONARIO (ID_FUNCIONARIO);

alter table DEPARTAMENTO
    add constraint FK_FACULTAD_IN_DEPT foreign key (ID_FACULTAD)
        references FACULTAD (ID_FACULTAD);

alter table DEPARTAMENTO
    add constraint FK_VICER_IN_DEPT foreign key (ID_VICERRECTORIA)
        references VICERRECTORIA (ID_VICERRECTORIA);

alter table FUNCIONARIO
    add constraint FK_PROGRAMA_IN_FUNCIONARIO foreign key (ID_FACULTAD, ID_PROGRAMA)
        references PROGRAMA (ID_FACULTAD, ID_PROGRAMA);

alter table DOCENTE
    add constraint FK_JEFE_IN_DOCENTE foreign key (ID_JEFE)
        references DOCENTE (ID_FUNCIONARIO);

alter table EDIFICIO
    add constraint FK_SEDE_IN_EDIFICIO foreign key (ID_SEDE)
        references SEDE (ID_SEDE);

alter table LOCACION
    add constraint FK_AUXILIAR_IN_LAB foreign key (ID_AUXILIAR)
        references FUNCIONARIO (ID_FUNCIONARIO);

alter table LABORATORIO

```

```

add constraint EK_EDIFICIO_IN_LAB foreign key (ID_EDIFICIO)
    references EDIFICIO (ID_EDIFICIO);

alter table LOCACION
    add constraint FK_AREA_IN_OFICINA foreign key (ID_AREA)
        references AREA (ID_AREA);

alter table LOCACION
    add constraint FK_EDIFICIO_IN_LOCACION foreign key (ID_EDIFICIO)
        references EDIFICIO (ID_EDIFICIO);

alter table PROCEDIMIENTO
    add constraint FK_FUN_ENTREGA_IN_PROC foreign key (FUNCIONARIO_ENTREGA)
        references FUNCIONARIO (ID_FUNCIONARIO);

alter table PROCEDIMIENTO
    add constraint FK_DOCENTE_ENTREGA_IN_PROC foreign key (FUNCIONARIO_ENTREGA)
        references DOCENTE (ID_FUNCIONARIO);

alter table PROCEDIMIENTO
    add constraint FK_FUNCIONARIO_IN_PROC foreign key (ID_AUXILIAR)
        references FUNCIONARIO (ID_FUNCIONARIO);

alter table PROCEDIMIENTO
    add constraint FK_ACTIVO_IN_PROC foreign key (ID_ACTIVO)
        references ACTIVO (ID_ACTIVO);

alter table PROGRAMA
    add constraint FK_FACULTAD_IN_PROGRAMA foreign key (ID_FACULTAD)
        references FACULTAD (ID_FACULTAD);

alter table SALON
    add constraint FK_EDIFICIO_IN_SALON foreign key (ID_EDIFICIO)
        references EDIFICIO (ID_EDIFICIO);

/*=====
/* CHECK TABLE FUNCIONARIO
*/
=====*/
alter table FUNCIONARIO
    add constraint FUNCIONARIO_CHK_TABLE
        CHECK (
            (TIPO_FUNCIONARIO = 'E'
                AND CARGO_EMPLEADO IS NOT NULL
                AND ID_AREA IS NOT NULL
                AND TITULO_DOCENTE IS NULL
                AND ROL_DOCENTE IS NULL
                AND ID_PROGRAMA IS NULL
                AND ID_FACULTAD IS NULL) OR
            (TIPO_FUNCIONARIO = 'D'
                AND CARGO_EMPLEADO IS NULL)
        );

```

```

        AND ID_AREA IS NULL
        AND TIPO_AUXILIAR IS NULL
        AND TITULO_DOCENTE IS NOT NULL
        AND ROL_DOCENTE IS NOT NULL
        AND ID_PROGRAMA IS NOT NULL
        AND ID_FACULTAD IS NOT NULL)
    );
/*=====
/* CHECK TABLE LOCACION
*/
/*=====*/
alter table LOCACION
add constraint LOCACION_CHK_TABLE
CHECK (
    (TIPO_LOCACION = 'S'
        AND CAPACIDAD_SALON IS NOT NULL
        AND PUESTOS_OFICINA IS NULL
        AND ID_AREA IS NULL
        AND ID_AUXILIAR IS NULL
        AND ACREDITACION_LAB IS NULL) OR
    (TIPO_LOCACION = 'O'
        AND CAPACIDAD_SALON IS NULL
        AND TIPO_SALON IS NULL
        AND PUESTOS_OFICINA IS NOT NULL
        AND ID_AREA IS NOT NULL
        AND ID_AUXILIAR IS NULL
        AND ACREDITACION_LAB IS NULL) OR
    (TIPO_LOCACION = 'L'
        AND CAPACIDAD_SALON IS NULL
        AND TIPO_SALON IS NULL
        AND PUESTOS_OFICINA IS NULL
        AND ID_AREA IS NULL
        AND ID_AUXILIAR IS NOT NULL
        AND ACREDITACION_LAB IS NOT NULL)
);

```

Resolución de arcos con restricciones de chequeo

Para el presente modelo, no es posible solucionar los arcos presentados con un supertipo, pues no es posible hacer una generalización entre las entidades FACULTAD y DEPARTAMENTO para eliminar el arco que forman con AREA y para las entidades FACULTAD y VICERRECTORÍA para eliminar el arco que forman con DEPARTAMENTO.

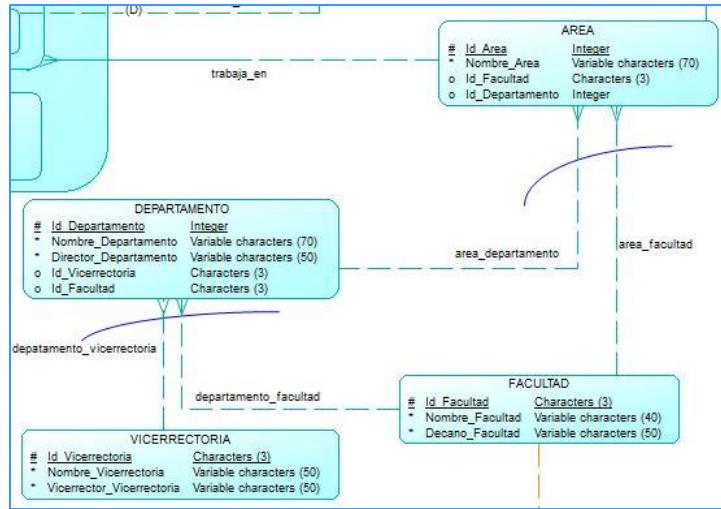


Figura 16 – ARCOS presentes en el diseño de la base de datos según se mostró en los diagramas generados en PowerDesigner

Se usará el siguiente código SQL para garantizar la integridad referencial entre estas relaciones.

```
alter table DEPARTAMENTO
add constraint DEPARTAMENTO_ARC_CHK
check (
    (id_vicerrectoria IS NOT NULL
     and id_facultad IS NULL) OR
    (id_vicerrectoria IS NULL
     and id_facultad IS NOT NULL)
);

alter table AREA
add constraint AREA_ARC_CHK
check (
    (id_facultad IS NOT NULL
     and id_departamento IS NULL) OR
    (id_facultad IS NULL
     and id_departamento IS NOT NULL)
);
```

Modificaciones adicionales al código de creación SQL de la base de datos

Debido a que la tabla PROCEDIMIENTO cuenta con un atributo llamado `tipo_procedimiento`, y en función del valor que tome ese atributo, algunos atributos son necesarios y otros no, se agregará una restricción de chequeo por medio de una alteración de tabla para esa estructura. El código a agregar es el siguiente:

```

alter table PROCEDIMIENTO
add constraint PROCEDIMIENTO_CHK_TABLE
CHECK (
    (TIPO_PROCEDIMIENTO = 'R'
     AND FUNCIONARIO_ENTREGA IS NULL) OR
    (TIPO_PROCEDIMIENTO = 'P'
     AND FUNCIONARIO_ENTREGA IS NULL) OR
    (TIPO_PROCEDIMIENTO = 'M'
     AND FUNCIONARIO_ENTREGA IS NULL) OR
    (TIPO_PROCEDIMIENTO = 'E'
     AND ESTADO_REVISION IS NULL
     AND FUNCIONARIO_ENTREGA IS NOT NULL)
);

```

Para la relación de cardinalidad 1-1 entre FUNCIONARIO y LOCACION, se generará una restricción de tipo UNIQUE que garantice que para cada laboratorio se asigne un auxiliar de laboratorio único. El código es el siguiente indicando en color verde la línea agregada:

```

/*=====
/* Table: LOCACION
*/
=====

create table LOCACION
(
    ID_EDIFICIO      INTEGER          not null,
    ID_LOCACION      INTEGER          not null,
    NOMBRE_LOCACION  VARCHAR2(100)    not null,
    TIPO_LOCACION    CHAR(1)         not null
        constraint CKC_TIPO_LOCACION check (TIPO_LOCACION in ('S','O','L')),

    CAPACIDAD_SALON  INTEGER          ,
    TIPO_SALON       CHAR(3)

        constraint CKC_TIPO_SALON check (TIPO_SALON is null or (TIPO_SALON in ('AUD','TAL','COM'))),

    PUESTOS_OFICINA  INTEGER          ,
    ID_AREA          INTEGER          ,
    ID_AUXILIAR      INTEGER          ,
    ACREDITACION_LAB VARCHAR2(20)    ,

    constraint PK_LOCACION primary key (ID_EDIFICIO, ID_LOCACION),
    constraint UK_LOCACION unique (ID_AUXILIAR)
);

```

Para la estructura que surge de la entidad intermedia PROCEDIMIENTO, se eliminarán los índices generados en PowerDesigner por las dos llaves foráneas que resultaron de la relación m-m entre las entidades AUXILIAR y ACTIVO en su momento. Esto es porque el motor de bases de datos genera un índice propio para esa estructura

y los dos índices adicionales generados resultan innecesarios. En color rojo aparece el código a eliminar del código de creación.

```
/*=====
/* Index: PROCEDIMIENTO2_FK                               */
/*=====
create index PROCEDIMIENTO2_FK on PROCEDIMIENTO (
    ID_AUXILIAR ASC
);
/*=====
/* Index: PROCEDIMIENTO_ACTIVO_FK                         */
/*=====
create index PROCEDIMIENTO_ACTIVO_FK on PROCEDIMIENTO (
    ID_ACTIVO ASC
);
```

También deben eliminarse las líneas que borran estos índices al inicio del código.

```
drop index PROCEDIMIENTO_ACTIVO_FK;
drop index PROCEDIMIENTO2_FK;
```

IMPLEMENTACIÓN DE LA BASE DE DATOS

La fase de implementación de la base de datos consiste únicamente en ejecutar el código de creación en un motor de bases de datos. Para este proyecto se escogió la versión de ORACLE 11g con el motor ORACLE APEX.

Una vez ejecutado el código en dicho motor, se procederá a generar el código de inserción de datos según las especificaciones dadas para el diseño y las restricciones de integridad por valor y referencial surgidas del diseño de la base de datos en PowerDesigner.

Ejecución del código de creación SQL en el motor de bases de datos

Se generaron los siguientes objetos dentro del motor:

The screenshot shows the Oracle APEX 'Explorador de Objetos' interface. On the left, under 'Tablas', there is a list of tables: ACTIVO, AREA, COMPRA, DEPARTAMENTO, EDIFICIO, FACULTAD, FUNCIONARIO, LOCACION, PROCEDIMIENTO, PROGRAMA, SEDE, and VICERRECTORIA. In the center, under 'Índices', a list of indexes is shown: ACTIVO_COMPRA_FK, AREA_DEPARTAMENTO_FK, AREA_FACULTAD_FK, AREA_OFICINA_FK, DEPARTAMENTO_FACULTAD_FK, DEPATAMENTO_VICERRECTORIA_FK, EDIFICIO_SEDE_FK, ENTREGA2_FK, LOCACION_EDIFICIO_FK, PK_ACTIVO, PK_AREA, PK_COMPRA, and PK_DEPARTAMENTO. On the right, a vertical list of constraints is displayed: PK_EDIFICIO, PK_FACULTAD, PK_FUNCIONARIO, PK_LOCACION, PK_PROCEDIMIENTO, PK_PROGRAMA, PK_SEDE, PK_VICERRECTORIA, PROGRAMA_FACULTAD_FK, SE_ENCUENTRA_EN2_FK, TRABAJA_EN_FK, UK_LOCACION, and USADO_POR2_FK.

Figura 17 – Objetos generados en ORACLE APEX al ejecutar el código de creación de la base de datos

Y se tienen las siguientes restricciones de integridad, que pueden ser consultadas usando el comando:

```
SELECT TABLE_NAME, CONSTRAINT_NAME, CONSTRAINT_TYPE, SEARCH_CONDITION FROM user_constraints ORDER BY TABLE_NAME, CONSTRAINT_NAME
```

TABLE_NAME	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION
ACTIVO	CKC_CATEGORIA_ACTIVO_ACTIVO	C	CATEGORIA_ACTIVO in ('CAE','DTS','LAB','LOG','SGT')
ACTIVO	FK_ACTIVO_USADO_POR_FUN	R	-
ACTIVO	FK_COMPRA_IN_ACTIVO	R	-
ACTIVO	FK_LOCACION_IN_ACTIVO	R	-
ACTIVO	PK_ACTIVO	P	-
ACTIVO	SYS_C0096531894	C	"ID_ACTIVO" IS NOT NULL
ACTIVO	SYS_C0096531895	C	"ID_COMPRA" IS NOT NULL
ACTIVO	SYS_C0096531896	C	"REFERENCIA_ACTIVO" IS NOT NULL
ACTIVO	SYS_C0096531897	C	"CATEGORIA_ACTIVO" IS NOT NULL
ACTIVO	SYS_C0096531898	C	"NOMBRE_ACTIVO" IS NOT NULL
ACTIVO	SYS_C0096531899	C	"MARCA_ACTIVO" IS NOT NULL
ACTIVO	SYS_C0096531900	C	"VALOR_ACTIVO" IS NOT NULL
ACTIVO	SYS_C0096531901	C	"DEPRECIACION_ACTIVO" IS NOT NULL
AREA	AREA_ARC_CHK	C	(id_facultad IS NOT NULL and id_departamento IS NULL) OR (id_facultad IS NULL and id_departamento IS NOT NULL)

AREA	FK_DEPARTAMENTO_IN_AREA	R	-
AREA	FK_FACULTAD_IN_AREA	R	-
AREA	PK_AREA	P	-
AREA	SYS_C0096531904	C	"ID_AREA" IS NOT NULL
AREA	SYS_C0096531905	C	"NOMBRE_AREA" IS NOT NULL
COMPRA	PK_COMPRA	P	-
COMPRA	SYS_C0096531914	C	"ID_COMPRA" IS NOT NULL
COMPRA	SYS_C0096531915	C	"FECHA_COMPRA" IS NOT NULL
COMPRA	SYS_C0096531916	C	"VALOR_COMPRA" IS NOT NULL
DEPARTAMENTO	DEPARTAMENTO_ARC_CHK	C	(id_vicerrectoria IS NOT NULL and id_facultad IS NULL) OR (id_vicerrectoria IS NULL and id_facultad IS NOT NULL)
DEPARTAMENTO	FK_FACULTAD_IN_DEPT	R	-
DEPARTAMENTO	FK_VICER_IN_DEPT	R	-
DEPARTAMENTO	PK_DEPARTAMENTO	P	-
DEPARTAMENTO	SYS_C0096531918	C	"ID_DEPARTAMENTO" IS NOT NULL
DEPARTAMENTO	SYS_C0096531919	C	"NOMBRE_DEPARTAMENTO" IS NOT NULL
DEPARTAMENTO	SYS_C0096531920	C	"DIRECTOR_DEPARTAMENTO" IS NOT NULL
EDIFICIO	FK_SEDE_IN_EDIFICIO	R	-
EDIFICIO	PK_EDIFICIO	P	-
EDIFICIO	SYS_C0096531922	C	"ID_SEDE" IS NOT NULL
EDIFICIO	SYS_C0096531923	C	"ID_EDIFICIO" IS NOT NULL
EDIFICIO	SYS_C0096531924	C	"NOMBRE_EDIFICIO" IS NOT NULL
FACULTAD	CKC_ID_FACULTAD_FACULTAD	C	ID_FACULTAD in ('AGR','JUR','ECO','ING','CIE')
FACULTAD	PK_FACULTAD	P	-
FACULTAD	SYS_C0096531926	C	"ID_FACULTAD" IS NOT NULL
FACULTAD	SYS_C0096531927	C	"NOMBRE_FACULTAD" IS NOT NULL
FACULTAD	SYS_C0096531928	C	"DECANO_FACULTAD" IS NOT NULL
FUNCIONARIO	CKC_TIPO_AUXILIAR_FUN	C	TIPO_AUXILIAR in ('INV','MAN','LAB')
FUNCIONARIO	CKC_TIPO_FUN	C	TIPO_FUNCIONARIO in ('E', 'D')
FUNCIONARIO	FK_AREA_IN_FUNCIONARIO	R	-

FUNCIONARIO	FK_JEFE_IN_FUNCIONARIO	R	-
FUNCIONARIO	FK_PROGRAMA_IN_FUNCIONARIO	R	-
FUNCIONARIO	FUNCIONARIO_CHK_TABLE	C	(TIPO_FUNCIONARIO = 'E' AND CARGO_EMPLEADO IS NOT NULL AND ID_AREA IS NOT NULL AND TITULO_DOCENTE IS NULL AND ROL_DOCENTE IS NULL AND ID_PROGRAMA IS NULL AND ID_FACULTAD IS NULL) OR (TIPO_FUNCIONARIO = 'D' AND CARGO_EMPLEADO IS NULL AND ID_AREA IS NULL AND TIPO_AUXILIAR IS NULL AND TITULO_DOCENTE IS NOT NULL AND ROL_DOCENTE IS NOT NULL AND ID_PROGRAMA IS NOT NULL AND ID_FACULTAD IS NOT NULL)
FUNCIONARIO	PK_FUNCIONARIO	P	-
FUNCIONARIO	SYS_C0096531907	C	"ID_FUNCIONARIO" IS NOT NULL
FUNCIONARIO	SYS_C0096531908	C	"NOMBRE_FUNCIONARIO" IS NOT NULL
FUNCIONARIO	SYS_C0096531909	C	"INIC_CONTRATO_FUNCIONARIO" IS NOT NULL
FUNCIONARIO	SYS_C0096531910	C	"TIPO_FUNCIONARIO" IS NOT NULL
LOCACION	CKC_TIPO_LOCACION	C	TIPO_LOCACION in ('S','O','L')
LOCACION	CKC_TIPO_SALON	C	TIPO_SALON is null or (TIPO_SALON in ('AUD','TAL','COM'))
LOCACION	FK_AREA_IN_OFICINA	R	-
LOCACION	FK_AUXILIAR_IN_LAB	R	-
LOCACION	FK_EDIFICIO_IN_LOCACION	R	-
LOCACION	LOCACION_CHK_TABLE	C	(TIPO_LOCACION = 'S' AND CAPACIDAD_SALON IS NOT NULL AND PUESTOS_OFICINA IS NULL AND ID_AREA IS NULL AND ID_AUXILIAR IS NULL AND ACREDITACION_LAB IS NULL) OR (TIPO_LOCACION = 'O' AND CAPACIDAD_SALON IS NULL AND TIPO_SALON IS NULL AND PUESTOS_OFICINA IS NOT NULL AND ID_AREA IS NOT NULL AND ID_AUXILIAR IS NULL AND ACREDITACION_LAB IS NULL) OR (TIPO_LOCACION = 'L' AND CAPACIDAD_SALON IS NULL AND TIPO_SALON IS NULL AND PUESTOS_OFICINA IS NULL AND ID_AREA IS NULL AND ID_AUXILIAR IS NOT NULL AND ACREDITACION_LAB IS NOT NULL)
LOCACION	PK_LOCACION	P	-
LOCACION	SYS_C0096531931	C	"ID_EDIFICIO" IS NOT NULL
LOCACION	SYS_C0096531932	C	"ID_LOCACION" IS NOT NULL
LOCACION	SYS_C0096531933	C	"NOMBRE_LOCACION" IS NOT NULL
LOCACION	SYS_C0096531934	C	"TIPO_LOCACION" IS NOT NULL
LOCACION	UK_LOCACION	U	-
PROCEDIMIENTO	CKC_ESTADO_REVISION_PROCEDIM	C	ESTADO_REVISION is null or (ESTADO_REVISION in ('R','M'))

PROCEDIMIENTO	CKC_TIPO_PROCEDIMIENT_PR OCEDIM	C	TIPO_PROCEDIMIENTO in ('E','P','R','M')
PROCEDIMIENTO	FK_ACTIVO_IN_PROC	R	-
PROCEDIMIENTO	FK_FUNCIONARIO_IN_PROC	R	-
PROCEDIMIENTO	FK_FUN_ENTREGA_IN_PROC	R	-
PROCEDIMIENTO	PK_PROCEDIMIENTO	P	-
PROCEDIMIENTO	PROCEDIMIENTO_CHK_TABLE	C	(TIPO_PROCEDIMIENTO = 'R' AND FUNCIONARIO_ENTREGA IS NULL) OR (TIPO_PROCEDIMIENTO = 'P' AND FUNCIONARIO_ENTREGA IS NULL) OR (TIPO_PROCEDIMIENTO = 'M' AND ESTADO_REVISION IS NULL AND FUNCIONARIO_ENTREGA IS NULL) OR (TIPO_PROCEDIMIENTO = 'E' AND ESTADO_REVISION IS NULL AND FUNCIONARIO_ENTREGA IS NOT NULL)
PROCEDIMIENTO	SYS_C0096531939	C	"ID_PROCE" IS NOT NULL
PROCEDIMIENTO	SYS_C0096531940	C	"ID_ACTIVO" IS NOT NULL
PROCEDIMIENTO	SYS_C0096531941	C	"ID_AUXILIAR" IS NOT NULL
PROCEDIMIENTO	SYS_C0096531942	C	"TIPO_PROCEDIMIENTO" IS NOT NULL
PROCEDIMIENTO	SYS_C0096531943	C	"FECHA_PROCEDIMIENTO" IS NOT NULL
PROGRAMA	FK_FACULTAD_IN_PROGRAMA	R	-
PROGRAMA	PK_PROGRAMA	P	-
PROGRAMA	SYS_C0096531947	C	"ID_FACULTAD" IS NOT NULL
PROGRAMA	SYS_C0096531948	C	"ID_PROGRAMA" IS NOT NULL
PROGRAMA	SYS_C0096531949	C	"NOMBRE_PROGRAMA" IS NOT NULL
SEDE	PK_SEDE	P	-
SEDE	SYS_C0096531951	C	"ID_SEDE" IS NOT NULL
SEDE	SYS_C0096531952	C	"NOMBRE_SEDE" IS NOT NULL
VICERRECTORIA	CKC_ID_VICERRECTORIA_VICE RREC	C	ID_VICERRECTORIA in ('FOR','INV','EXT','ADM')
VICERRECTORIA	PK_VICERRECTORIA	P	-
VICERRECTORIA	SYS_C0096531954	C	"ID_VICERRECTORIA" IS NOT NULL
VICERRECTORIA	SYS_C0096531955	C	"NOMBRE_VICERRECTORIA" IS NOT NULL
VICERRECTORIA	SYS_C0096531956	C	"VICERECTOR_VICERRECTORIA" IS NOT NULL

Tabla 13 - Restricciones registradas en ORACLE APEX tras la ejecución del código de creación de la base de datos

Generación de código SQL para la inserción de datos en el esquema de la base de datos

El código SQL para insertar datos en la base de datos se generó por medio de un script escrito en el lenguaje programación Java, aprovechando sus características como lenguaje para la programación orientada a objetos.

En resumen, se creó una clase por cada una de las tablas de la base de datos. Se agregaron los atributos correspondientes para cada una, donde las llaves foráneas se tomaron como referencias a otros objetos de las clases de las otras estructuras.

A continuación un ejemplo:

```
class Activo {  
  
    private int id;  
    private Compra compra;  
    private String ref;  
    private String categoria;  
    private String nombre;  
    private String marca;  
    private double valor;  
    private int depreciacion;  
    private Funcionario funcionario;  
    private Edificio edificio;  
    private Locacion locacion;  
    private String material;  
    private String color;  
    private String tamaño;  
    private String componentes;
```

Figura 18 – Definición de una tabla de la base de datos como clase como metodología para la generación del código de inserción SQL

De esta manera se fueron instanciando objetos por medio de ciclos y utilizando algunos constructores ideados para ciertos registros. Este es un ejemplo de uno para generar objetos que se convertirán en registros de Televisores:

```
public Activo(int id, Compra c, Edificio e, Locacion l) {  
  
    // Constructor para Televisores  
    this.id = id;  
    this.compra = c;  
    this.ref = randomAlphaNumeric(15);  
    this.categoría = "CAE";  
    this.nombre = "Televisor";  
    this.tamaño = Integer.toString((getRandomInt(28, 32))) + " pulgadas";  
    this.marca = getRandomMarca(2);  
    this.valor = getRandomInt(1000000, 1500000);  
    this.depreciacion = 10;  
    this.edificio = e;  
    this.locacion = l;  
}
```

Figura 19 – Ejemplo de constructor que permite generar registros para una tabla definida como clase usando el lenguaje de programación Java

Se usaron algunos métodos de apoyo para generar valores aleatorios, entre ellos un método para generar enteros, para los activos: marcas, materiales, tamaños, colores y para los funcionarios nombres con sus apellidos.

```
public static String getRandomColor(int p) {
    String [] colores_escritorios = {"Marrón", "Beige", "Blanco", "Gris"};
    String [] colores_giratorias = {"Negro", "Azul Oscuro", "Gris"};
    if (p == 1) {
        return colores_escritorios[getInt(0, 3)];
    }
    if (p == 4) {
        return colores_giratorias[getInt(0, 3)];
    }
    return "null";
}
```

Figura 20 – Ejemplo de método implementado para generar valores aleatorios para ciertos registros de la base de datos

Los registros de las siguientes tablas se generaron según los valores de los dominios especificados durante el diseño:

VICERRECTORIA, DEPARTAMENTO, FACULTAD, AREA, PROGRAMA, SEDE y EDIFICIO

En cuanto a las siguientes tablas, se generaron registros aleatorios guardando las condiciones de los dominios que aparecen en la *tabla 12* del presente documento

COMPRA, ACTIVO, LOCACION y FUNCIONARIO

El número de registros generados para ACTIVO, LOCACION (excepto los laboratorios) y FUNCIONARIO puede variar cada vez que se ejecute el script. Eso significa que no siempre se obtendrán 610 funcionarios con 80 de ellos docentes, la misma cantidad de activos ni de salones u oficinas.

Cada clase cuenta con un método que imprime una sentencia de inserción por cada tipo de constructor, tal como se muestra a continuación:

```
public void printInfo_d() {
    System.out.println("insert into FUNCIONARIO "
        + "(ID_FUNCIONARIO, NOMBRE_FUNCIONARIO, INIC_CONTRATO_FUNCIONARIO, FIN_CONTRATO_FUNCIONARIO, "
        + " ID_JEFE, TIPO_FUNCIONARIO, CARGO_EMPLEADO, ID_AREA, TITULO_DOCENTE, ROL_DOCENTE, ID_FACULTAD, "
        + " ID_PROGRAMA, TIPO_AUXILIAR) "
        + "values (" + this.id + ",\''" + this.nombre + "\',TO_DATE(\''" + this.inic + "\', \\'dd/mm/yyyy\'), "
        + "null, null,\''" + this.tipo + "\', null, null,\''"
        + this.titulo + "\',\''" + this.rol + "\',\''" + this.facultad.getId() + "\',\''"
        + this.programa.getId() + "\', null);");
    System.out.println();
}
```

Figura 21 – Ejemplo de método implementado para generar una sentencia de inserción SQL

El anterior método imprime una sentencia de inserción de un registro de docente en la tabla FUNCIONARIO.

Ejecución del código de inserción SQL en el motor de bases de datos

Para ejecutar el código de inserción en el motor de bases de datos, se necesitó dividirlo en archivos más pequeños, es decir con menor número de sentencias para que sea aceptado por ORACLE APEX, pues esta herramienta impone un tamaño límite por archivo que ronda los 500 KB aproximadamente.

Así se dividió el código de inserción generado del script en Java en seis diferentes archivos que se ejecutaron en el motor, tal como se muestra a continuación:

Nombre	Creado	Actualizado por	Actualizado	Bytes	Resultados	Ejecutar
ActivosFijos_Creation_v_2_2	Hace 11 días	DIQUINTERO@UNAL.EDU.CO	Hace 11 días	19.252	2	▶
AF_Data_v_2_4_a	Hace 3 días	DIQUINTERO@UNAL.EDU.CO	Hace 3 días	163.946	1	▶
AF_Data_v_2_4_b	Hace 3 días	DIQUINTERO@UNAL.EDU.CO	Hace 3 días	229.747	1	▶
AF_Data_v_2_4_c	Hace 3 días	DIQUINTERO@UNAL.EDU.CO	Hace 3 días	259.458	1	▶
AF_Data_v_2_4_d	Hace 3 días	DIQUINTERO@UNAL.EDU.CO	Hace 3 días	398.834	1	▶
AF_Data_v_2_4_e	Hace 3 días	DIQUINTERO@UNAL.EDU.CO	Hace 3 días	307.631	1	▶
AF_Data_v_2_4_f	Hace 3 días	DIQUINTERO@UNAL.EDU.CO	Hace 3 días	152.734	1	▶

Figura 22 – Subdivisión del código de inserción para poder ser cargado en ORACLE APEX según las limitaciones de este motor

La letra al final de los archivos AF_Data_v_2_4 indica el orden en que se ejecutaron estos seis archivos de inserción comenzando desde el archivo con letra ‘a’ hasta el archivo con letra ‘f’, siguiendo el orden del abecedario.

Luego de haber generado y ejecutado los códigos de inserción SQL, se encontraron algunos errores. Por ejemplo, cuando se escribió el script en Java, no se consideraron los tipos de salón: Taller y Salas de Cómputo, y se generaron únicamente salones de clase normales y Auditorios.

Hay que tener en cuenta además que la capacidad de las salas de cómputo está determinada por los computadores que se encuentran en dichos salones. Para solucionar este problema, se ubicaron los salones que contaban con computadores usando la siguiente consulta:

```
SELECT ID_EDIFICIO, ID_LOCACION, COUNT(*) FROM ACTIVO NATURAL JOIN LOCACION WHERE NOMBRE_ACTIVO = 'Computador de Escritorio' AND TIPO_LOCACION = 'S' GROUP BY ID_EDIFICIO, ID_LOCACION
```

Se tomó nota de cada uno de estos salones como del número de computadores, para luego usar un comando de actualización sobre cada uno de estos registros, así:

```
UPDATE LOCACION
SET CAPACIDAD_SALON = ##, TIPO_SALON = 'COM'
WHERE ID_EDIFICIO = ## AND ID_LOCACION = ##
```

Así por cada salón se ejecutó uno de estos comandos de actualización, cambiando las identificaciones del mismo en WHERE y definiendo la capacidad de cada uno según el número de computadores encontrado en el mismo con el primer comando.

Contando con estas salas de cómputo registradas debidamente en la base de datos, se procedió a generar algunos talleres con el siguiente comando:

```
UPDATE LOCACION
SET TIPO_SALON = 'TAL'
WHERE MOD(ID_LOCACION, 25) = 0 AND TIPO_LOCACION = 'S' AND TIPO_SALON IS NULL
```

De esta manera se decidió de manera arbitraria que los salones de clase cuya identificación fuera 25 o 50 se cambiaran a talleres.

Validación de la inserción de datos en la base de datos ejecutando consultas de SQL

A continuación se presentan 18 consultas de ejemplo sobre la base de datos para verificar la funcionalidad de la base de datos según fue diseñada y los registros insertados.

Para cada consulta se presenta su equivalente en álgebra relacional y en SQL, a excepción de algunas consultas de funciones agregadas y de proyección generalizada que por su extensión, no conviene expresarlas en álgebra relacional. También se creará una vista para poder consultar los resultados en el motor de bases de datos cuando se desee.

Consulta 1: Selección

Desplegar los auditorios, talleres y salas de cómputo de la universidad con todo su esquema.

$\sigma_{TIPO_SALON \neq NULL}(LOCACION)$

```
CREATE VIEW consulta_1 AS
SELECT * FROM LOCACION WHERE TIPO_SALON IS NOT NULL ORDER BY TIPO_SALON, ID_EDIFICIO, ID_LOCACION
```

Consulta 2: Proyección

Desplegar la identificación, el nombre, el título y el identificador de programa de los docentes titulares de la universidad.

$\Pi_{ID_FUNCIONARIO, NOMBRE_FUNCIONARIO, TITULO_DOCENTE, ID_PROGRAMA} (\sigma_{ROL_DOCENTE='Titular'}(FUNCIONARIO))$

```
CREATE VIEW consulta_2 AS
SELECT ID_FUNCIONARIO, NOMBRE_FUNCIONARIO, TITULO_DOCENTE, ID_PROGRAMA
FROM FUNCIONARIO
WHERE ROL_DOCENTE = 'Titular'
ORDER BY ID_PROGRAMA, TITULO_DOCENTE
```

Consulta 3: Producto Natural basado en Producto Cartesiano

Desplegar la identificación y el nombre de los jefes de área indicando el nombre del área de la cual son jefes.

$$\Pi_{ID_FUNCIONARIO, NOMBRE_FUNCIONARIO, NOMBRE_AREA} \left(\sigma_{(F.ID_AREA=A.ID_AREA) \wedge (CARGO_EMPLEADO='Jefe de Área')} (\rho_F(FUNCIONARIO) \times \rho_A(AREA)) \right)$$

```
CREATE VIEW consulta_3 AS
SELECT ID_FUNCIONARIO, NOMBRE_FUNCIONARIO, NOMBRE_AREA
FROM FUNCIONARIO F, AREA A
WHERE (F.ID_AREA = A.ID_AREA) AND
      CARGO_EMPLEADO = 'Jefe de Área'
```

Consulta 4: Producto Natural

Desplegar el identificador, la referencia, el tamaño y la ubicación de los televisores de marca Sony de la sede principal de la universidad.

$$\Pi_{ID_ACTIVO, REFERENCIA_ACTIVO, TAMANO_ACTIVO, NOMBRE_LOCACION} \left(\sigma_{(NOMBRE_ACTIVO='Televisor') \wedge (ACTIVO \bowtie_\theta LOCACION \bowtie_\theta EDIFICIO \bowtie_\theta SEDE) \wedge (NOMBRE_ACTIVO='Sony')} \wedge (ID_SEDE=1) \right)$$

```
CREATE VIEW consulta_4 AS
SELECT ID_ACTIVO, REFERENCIA_ACTIVO, TAMANO_ACTIVO, NOMBRE_LOCACION
FROM ACTIVO NATURAL JOIN LOCACION NATURAL JOIN EDIFICIO NATURAL JOIN SEDE
WHERE NOMBRE_ACTIVO = 'Televisor' AND MARCA_ACTIVO = 'Sony' AND ID_SEDE = 1
ORDER BY NOMBRE_LOCACION
```

Consulta 5: Unión

Desplegar el nombre de todos los directivos de la universidad, es decir, vicerrectores, decanos y directores, indicando la entidad y el nombre de la entidad a la que pertenecen.

$$\rho_{NOMBRE_DIRECTIVO, ENTIDAD, NOMBRE_ENTIDAD} \\
\left(\left(\Pi_{VICERRECTOR_VICERRECTORIA, 'Vicerrectoría', NOMBRE_VICERRECTORIA}(VICERRECTORIA) \right) \cup \right. \\
\left. \left(\Pi_{DECANO_FACULTAD, 'Facultad', NOMBRE_FACULTAD}(FACULTAD) \right) \cup \right. \\
\left. \left(\Pi_{DIRECTOR_DEPARTAMENTO, 'Departamento', NOMBRE_DEPARTAMENTO}(DEPARTAMENTO) \right) \right)$$

```
CREATE VIEW consulta_5 AS
SELECT VICERRECTOR_VICERRECTORIA NOMBRES_DIRECTIVOS, 'Vicerrectoría' ENTIDAD, NOMBRE_VICERRECTORIA
NOMBRE_ENTIDAD FROM VICERRECTORIA UNION
(SELECT DECANO_FACULTAD, 'Facultad', NOMBRE_FACULTAD FROM FACULTAD) UNION
(SELECT DIRECTOR_DEPARTAMENTO, 'Departamento', NOMBRE_DEPARTAMENTO FROM DEPARTAMENTO)
```

Consulta 6: Diferencia

Desplegar la identificación y el nombre de los funcionarios a los cuales no se les ha hecho entrega formal de su puesto de trabajo.

$$\Pi_{ID_FUNCIONARIO, NOMBRE_FUNCIONARIO}(FUNCIONARIO) \\
-\Pi_{ID_FUNCIONARIO, NOMBRE_FUNCIONARIO}(ACTIVO \bowtie_\theta FUNCIONARIO)$$

```
CREATE VIEW consulta_6 AS
SELECT ID_FUNCIONARIO, NOMBRE_FUNCIONARIO FROM FUNCIONARIO MINUS
SELECT ID_FUNCIONARIO, NOMBRE_FUNCIONARIO FROM ACTIVO NATURAL JOIN FUNCIONARIO
```

Consulta 7: Intersección

Desplegar los talleres que cuentan con un pupitre para el docente y un televisor al mismo tiempo, indicando su identificación (llave compuesta ID_EDIFICIO, ID_LOCACION) y capacidad.

$$\Pi_{ID_EDIFICIO, ID_LOCACION, NOMBRE_LOCACION, CAPACIDAD_SALON} \left(\sigma_{(NOMBRE_ACTIVO='Pupitre') \wedge (TIPO_SALON='TAL')} (LOCACION \bowtie_\theta ACTIVO) \right) \cap$$

$$\Pi_{ID_EDIFICIO, ID_LOCACION, NOMBRE_LOCACION, CAPACIDAD_SALON} \left(\sigma_{(NOMBRE_ACTIVO='Televisor') \wedge (TIPO_SALON='TAL')} (LOCACION \bowtie_\theta ACTIVO) \right)$$

```
CREATE VIEW consulta_7 AS
SELECT ID_EDIFICIO, ID_LOCACION, NOMBRE_LOCACION, CAPACIDAD_SALON FROM LOCACION NATURAL JOIN ACTIVO
WHERE NOMBRE_ACTIVO = 'Pupitre' AND TIPO_SALON = 'TAL' INTERSECT
(SELECT ID_EDIFICIO, ID_LOCACION, NOMBRE_LOCACION, CAPACIDAD_SALON FROM LOCACION NATURAL JOIN ACTIVO
WHERE NOMBRE_ACTIVO = 'Televisor' AND TIPO_SALON = 'TAL')
```

Consulta 8: División

Desplegar todo el nombre de las salas de computación en las cuales hay computadores de todas las marcas de computadores existentes en el inventario.

Se harán dos asignaciones; una para el dividendo y otra para el divisor:

$$\text{divisor} \leftarrow \Pi_{\text{MARCA_ACTIVO}} \left(\sigma_{\text{NOMBRE_ACTIVO}='Computador de Escritorio'} (\text{ACTIVO}) \right)$$

```
CREATE VIEW divisor AS
SELECT DISTINCT MARCA_ACTIVO FROM ACTIVO WHERE NOMBRE_ACTIVO = 'Computador de Escritorio'
```

$$\text{dividendo} \leftarrow \Pi_{\text{NOMBRE_LOCACION}, \text{MARCA_ACTIVO}} \left(\sigma_{\text{TIPO_SALON}='COM'} (\text{LOCACION} \bowtie_\theta \text{ACTIVO}) \right)$$

```
CREATE VIEW dividendo AS
SELECT NOMBRE_LOCACION, MARCA_ACTIVO FROM LOCACION NATURAL JOIN ACTIVO WHERE TIPO_SALON = 'COM'
```

$$\Pi_{\text{NOMBRE_LOCACION}}(\text{dividendo}) - \Pi_{\text{NOMBRE_LOCACION}} \left((\Pi_{\text{NOMBRE_LOCACION}}(\text{dividendo}) \times \text{divisor}) - \text{dividendo} \right)$$

```
CREATE VIEW consulta_8 AS
SELECT NOMBRE_LOCACION FROM dividendo MINUS (SELECT NOMBRE_LOCACION FROM (SELECT * FROM (SELECT NOMBRE_LOCACION FROM dividendo), divisor MINUS (SELECT * FROM dividendo)))
```

Consulta 9: Proyección Generalizada

Desplegar el código de cada compra sin la parte inicial que indica el año y presentar la fecha subdividida en día, mes y año.

$$\rho(\text{cod_compra}, \text{año_compra}, \text{mes_compra}, \text{dia_compra}) \left(\begin{array}{l} \Pi_{\text{TO_NUMBER}(\text{SUBSTR}(\text{TO_CHAR}(\text{id_compra}), 5, 3)), (\text{COMPRA})} \\ \text{TO_CHAR}(\text{fecha_compra}, 'yyyy'), \\ \text{TO_CHAR}(\text{fecha_compra}, 'mm'), \\ \text{TO_CHAR}(\text{fecha_compra}, 'dd'), \\ \text{valor_compra} \end{array} \right)$$

```
CREATE VIEW consulta_9 AS
SELECT TO_NUMBER(SUBSTR(TO_CHAR(id_compra), 5, 3)) cod_compra,
       TO_CHAR(fecha_compra, 'yyyy') año_compra,
       TO_CHAR(fecha_compra, 'mm') mes_compra,
       TO_CHAR(fecha_compra, 'dd') dia_compra,
       valor_compra
FROM COMPRA
```

Consulta 10: Proyección Generalizada

Desplegar el nombre de los activos que hacen parte de las secciones de laboratorios indicando el porcentaje de su valor con respecto al de la compra (en la cual se incluyeron otros activos de otras áreas). Mostrar el valor del activo y de la compra con formato de moneda.

```
CREATE VIEW consulta_10 AS
SELECT NOMBRE_ACTIVO,
       (ROUND((VALOR_ACTIVO * 100)/VALOR_COMPRA, 2) || '%') PORCENTAJE_COMPRA,
       (TO_CHAR(VALOR_ACTIVO, '$999,999,999.99') || ' COP') VALOR_ACTIVO,
       (TO_CHAR(VALOR_COMPRA, '$999,999,999.99') || ' COP') VALOR_COMPRA
FROM ACTIVO NATURAL JOIN COMPRA
WHERE CATEGORIA_ACTIVO = 'LAB'
```

Consulta 11: Proyección Generalizada

Desplegar el esquema de la tabla DEPARTAMENTO pero con los nombres de los directores subdivididos en primer nombre, segundo nombre, primer apellido y segundo apellido. Algunos de estos directores tienen solo el primer apellido registrado en la base de datos.

```
CREATE VIEW consulta_11 AS
SELECT id_departamento,
       nombre_departamento,
       REGEXP_SUBSTR(director_departamento, '(\w+)\s', 1) primer_nombre_direc,
       REGEXP_SUBSTR(director_departamento, '(\w+)\s', INSTR(director_departamento, ' ', 1))
       segundo_nombre_direc,
       REGEXP_SUBSTR(director_departamento, '(\w+)\s', INSTR(director_departamento, ' ', INSTR(director_departamento, ' ', 1)+1)) primer_apellido_direc,
```

```

        SUBSTR(director_departamento, INSTR(director_departamento, ' ',
INSTR(director_departamento, ' ', 1, 2)+1)) segundo_apellido_direc,
        id_vicerrectoria,
        id_facultad

FROM departamento WHERE (length(director_departamento) - length(replace(director_departamento, ' ',
'')) +1) = 4

UNION

SELECT id_departamento,
        nombre_departamento,
        REGEXP_SUBSTR(director_departamento, '(\w+)\s', 1) primer_nombre_direc,
        REGEXP_SUBSTR(director_departamento, '(\w+)\s', INSTR(director_departamento, ' ', 1))
segundo_nombre_direc,
        SUBSTR(director_departamento, INSTR(director_departamento, ' ',
INSTR(director_departamento, ' ', 1)+1)) primer_apellido_direc,
        null segundo_apellido_direc,
        id_vicerrectoria,
        id_facultad

FROM departamento WHERE (length(director_departamento) - length(replace(director_departamento, ' ',
'')) +1) = 3

```

Consulta 12: Funciones Agregadas

Desplegar el número de funcionarios por área a los cuales no se les ha hecho entrega formal de su puesto de trabajo.

```

CREATE VIEW consulta_12 AS

SELECT NOMBRE_AREA, COUNT(*) NUM_SIN_ENTREGA

FROM (SELECT ID_FUNCIONARIO, NOMBRE_FUNCIONARIO, ID_AREA FROM FUNCIONARIO MINUS

      SELECT ID_FUNCIONARIO, NOMBRE_FUNCIONARIO, ID_AREA FROM ACTIVO NATURAL JOIN FUNCIONARIO)

NATURAL JOIN AREA

GROUP BY NOMBRE_AREA

ORDER BY NUM_SIN_ENTREGA desc

```

Consulta 13: Funciones Agregadas, with clause, Inner Join

Indicar el número de puestos entregados por cargo, junto con el número de empleados por cargo y el porcentaje de avance en la entrega.

```

CREATE VIEW consulta_13 AS

WITH puestos_asignados AS

(SELECT CARGO_EMPLEADO, COUNT(CARGO_EMPLEADO) PUESTOS_ASIGNADOS

  FROM (SELECT ID_FUNCIONARIO FROM ACTIVO WHERE ID_FUNCIONARIO IS NOT NULL GROUP BY ID_FUNCIONARIO)

NATURAL JOIN FUNCIONARIO WHERE CARGO_EMPLEADO IS NOT NULL GROUP BY CARGO_EMPLEADO),

empleados_por_cargo AS

(SELECT CARGO_EMPLEADO, COUNT(*) EMPLEADOS_EXISTENTES

```

```

FROM FUNCIONARIO WHERE CARGO_EMPLEADO IS NOT NULL GROUP BY CARGO_EMPLEADO)

SELECT puestos_asignados.CARGO_EMPLEADO,
       PUESTOS_ASIGNADOS,
       EMPLEADOS_EXISTENTES,
       (ROUND((PUESTOS_ASIGNADOS*100)/EMPLEADOS_EXISTENTES, 2) || '%') PORCENTAJE_ASIGNADO

FROM puestos_asignados INNER JOIN empleados_por_cargo
    ON puestos_asignados.CARGO_EMPLEADO = empleados_por_cargo.CARGO_EMPLEADO

```

Consulta 14: with clause, Right Outer Join

Desplegar una tabla en la que se concatenen dos relaciones. Una en la que se muestren los funcionarios a los cuales ya se les asignó un computador y otra que muestre los funcionarios a los que ya se les asignó un escritorio. Mostrar las equivalencias de ambas relaciones con los valores nulos al lado derecho.

```

CREATE OR REPLACE VIEW consulta_14 AS

WITH funcionarios_con_computador AS (SELECT ID_FUNCIONARIO ID_FUNCIONARIO_1, NOMBRE_FUNCIONARIO
NOMBRE_FUNCIONARIO_1, ID_ACTIVO COMPUTADOR FROM FUNCIONARIO NATURAL JOIN ACTIVO WHERE NOMBRE_ACTIVO =
'Computador de Escritorio'),

funcionarios_con_escritorio AS (SELECT ID_FUNCIONARIO ID_FUNCIONARIO_2, NOMBRE_FUNCIONARIO
NOMBRE_FUNCIONARIO_2, ID_ACTIVO ESCRITORIO FROM FUNCIONARIO NATURAL JOIN ACTIVO WHERE NOMBRE_ACTIVO =
'Escritorio')

SELECT * FROM funcionarios_con_computador RIGHT OUTER JOIN funcionarios_con_escritorio ON
funcionarios_con_computador.ID_FUNCIONARIO_1 = funcionarios_con_escritorio.ID_FUNCIONARIO_2

ORDER BY COMPUTADOR desc

```

Debido a que los puestos se habían asignado completos, es decir, computador, escritorio y silla. Se eliminaron actualizaron algunas tuplas para hacer la consulta anterior, con el siguiente comando:

```

UPDATE ACTIVO
SET ID_FUNCIONARIO = NULL
WHERE MOD(ID_FUNCIONARIO, 40) = 0 AND NOMBRE_ACTIVO = 'Computador de Escritorio'

```

Consulta 15: Consultas Anidadas

Desplegar la identificación y el nombre de los funcionarios a los cuales no se les ha entregado puesto de trabajo.

```

CREATE VIEW consulta_15 AS

SELECT ID_FUNCIONARIO, NOMBRE_FUNCIONARIO FROM FUNCIONARIO WHERE ID_FUNCIONARIO NOT IN (SELECT
ID_FUNCIONARIO FROM FUNCIONARIO NATURAL JOIN ACTIVO)

```

Consulta 16: Consultas Anidadas

Verificar que los valores de los activos que pertenecen a una misma compra no superan el valor total de la compra. De ser así, desplegar los datos de la compra.

Para asegurarse de que no haya totales mayores al valor de una compra se usó la siguiente sentencia de actualización:

```
UPDATE COMPRA
```

```

SET VALOR_COMPRA = (WITH sumas_valores_activos AS (SELECT ID_COMPRA, SUM(VALOR_ACTIVO)
SUMA_VALORES_ACTIVOS FROM ACTIVO GROUP BY ID_COMPRA)

SELECT sva.SUMA_VALORES_ACTIVOS FROM sumas_valores_activos sva WHERE sva.ID_COMPRA =
COMPRA.ID_COMPRA)

```

La siguiente consulta permite verificar que se obtuvo el resultado correcto:

```

WITH a AS
(SELECT ID_COMPRA, SUM(VALOR_ACTIVO) FROM ACTIVO GROUP BY ID_COMPRA),
b AS
(SELECT ID_COMPRA, VALOR_COMPRA FROM COMPRA)
SELECT * FROM a INNER JOIN b ON a.ID_COMPRA = b.ID_COMPRA

```

Ahora modificamos deliberadamente dos registros de la tabla COMPRA

```

UPDATE COMPRA
SET VALOR_COMPRA = VALOR_COMPRA - 245000
WHERE ID_COMPRA = 2020041
UPDATE COMPRA
SET VALOR_COMPRA = VALOR_COMPRA + 500000
WHERE ID_COMPRA = 2020033

```

La consulta es la siguiente:

```

CREATE VIEW consulta_16 AS
WITH sumas AS (SELECT ID_COMPRA, SUM(VALOR_ACTIVO) SUMA FROM ACTIVO GROUP BY ID_COMPRA)
SELECT ID_COMPRA, SUMA, (SUMA - (SELECT VALOR_COMPRA FROM COMPRA C WHERE C.ID_COMPRA =
sumas.ID_COMPRA)) EXCESO
FROM sumas
WHERE SUMA > (SELECT VALOR_COMPRA FROM COMPRA WHERE ID_COMPRA = sumas.ID_COMPRA)

```

La siguiente consulta busca las compras cuyo valor es menor al que apareció en la factura:

```

WITH sumas AS (SELECT ID_COMPRA, SUM(VALOR_ACTIVO) SUMA FROM ACTIVO GROUP BY ID_COMPRA)
SELECT ID_COMPRA, SUMA, ((SELECT VALOR_COMPRA FROM COMPRA C WHERE C.ID_COMPRA = sumas.ID_COMPRA)-SUMA) FALTANTE
FROM sumas
WHERE SUMA < (SELECT VALOR_COMPRA FROM COMPRA WHERE ID_COMPRA = sumas.ID_COMPRA)

```

Consulta 17: Consultas Anidadas

Desplegar el detalle de las compras realizadas que han superado el promedio del valor de compras en más de 25 millones de pesos.

```

CREATE VIEW consulta_17 AS
SELECT ID_COMPRA, VALOR_COMPRA, FECHA_COMPRA, ID_ACTIVO, NOMBRE_ACTIVO, VALOR_ACTIVO FROM COMPRA
NATURAL JOIN ACTIVO WHERE (VALOR_COMPRA - (SELECT AVG(VALOR_COMPRA) FROM COMPRA)) >= 25000000 ORDER BY
ID_COMPRA

```

Consulta 18: Consultas Anidadas, Proyección Generalizada

Desplegar una relación en la cual se muestre el identificador, el nombre, el valor por el cual se compró cada activo, la fecha de la compra, el índice de depreciación, los días transcurridos desde la compra, el índice de depreciación acumulado y el valor que tiene hoy cada activo

```
CREATE VIEW consulta_18 AS
SELECT S.* , ROUND(VALOR_ACTIVO - (VALOR_ACTIVO*S.DEPRECIACION_TRANSURRIDA)/100, 0) VALOR_ACTIVO_HOY FROM
(SELECT SQ.* , ROUND(((SQ.DIAS_TRANSURRIDOS * DEPRECIACION_ACTIVO)/365), 2) DEPRECIACION_TRANSURRIDA
FROM
(SELECT ID_ACTIVO, NOMBRE_ACTIVO, VALOR_ACTIVO, FECHA_COMPRA, DEPRECIACION_ACTIVO, ROUND(SYSDATE -
FECHA_COMPRA, 0) DIAS_TRANSURRIDOS FROM ACTIVO NATURAL JOIN COMPRA) SQ) S
```

Implementación de PL/SQL

A continuación se presenta una serie de procedimientos, funciones y disparadores agregados a la base de datos como implementación del lenguaje PL/SQL.

Procedimientos

El siguiente procedimiento imprime una factura que contiene el detalle de todos los activos comprados con una misma factura cuyo ID_COMPRA se evalúa como parámetro del procedimiento.

```
create or replace PROCEDURE generarFactura(ID_COM IN NUMBER)
IS
    CURSOR activos IS SELECT ID_ACTIVO, REFERENCIA_ACTIVO, NOMBRE_ACTIVO, MARCA_ACTIVO, VALOR_ACTIVO FROM ACTIVO WHERE
ID_COMPRA = ID_COM ORDER BY NOMBRE_ACTIVO, ID_ACTIVO;
    total NUMBER;
    suma NUMBER;
    reg activos%ROWTYPE;
    fecha DATE;
BEGIN
    SELECT VALOR_COMPRA INTO total FROM COMPRA WHERE ID_COMPRA = ID_COM;
    SELECT SUM(VALOR_ACTIVO) INTO suma FROM ACTIVO WHERE ID_COMPRA = ID_COM;
    SELECT FECHA_COMPRA INTO fecha FROM COMPRA WHERE ID_COMPRA = ID_COM;
    dbms_output.put_line('');
    dbms_output.put_line('FECHA DE COMPRA: ' || fecha);
    dbms_output.put_line('');
    dbms_output.put_line('ID FACTURA: ' || ID_COM);
    dbms_output.put_line('-----');
    dbms_output.put_line(RPAD('ID Activo', 15) || RPAD('REFERENCIA', 25) || RPAD('NOMBRE', 30) || RPAD('MARCA', 17) ||
'VALOR');
    dbms_output.put_line('-----');
    OPEN activos;
LOOP
    FETCH activos INTO reg;
    EXIT WHEN activos%NOTFOUND;
```

```

        dbms_output.put_line(RPAD(TO_CHAR(reg.ID_ACTIVO), 15) || RPAD(TO_CHAR(reg.REFERENCIA_ACTIVO), 25) ||
RPAD(TO_CHAR(reg.NOMBRE_ACTIVO), 30) ||
' COP');

      END LOOP;

      dbms_output.put_line('-----');
      dbms_output.put_line('Suma: ' || TO_CHAR(suma, '$999,999,999.00') || ' COP');
      dbms_output.put_line('.....');
      dbms_output.put_line('Valor Registrado en Factura: ' || TO_CHAR(total, '$999,999,999.00') || ' COP');
      dbms_output.put_line('.....');
      dbms_output.put_line('Desfase: ' || TO_CHAR(total-suma, '$999,999,999.00') || ' COP');

      CLOSE activos;
END;

```

Otro procedimiento implementado, imprime un acta de entrega de puesto de trabajo en formato reconocible para un documento de HTML:

```

create or replace PROCEDURE genActaEntrega(ID_FUN IN NUMBER)
IS
  CURSOR activos IS SELECT ID_ACTIVO, REFERENCIA_ACTIVO, NOMBRE_ACTIVO, MARCA_ACTIVO, MATERIAL_ACTIVO, COLOR_ACTIVO,
TAMANO_ACTIVO, COMPONENTES_ACTIVO
    FROM ACTIVO NATURAL JOIN FUNCIONARIO
    WHERE ID_FUNCIONARIO = ID_FUN ORDER BY NOMBRE_ACTIVO, ID_ACTIVO;
  reg activos%ROWTYPE;
  fecha DATE;
  CURSOR auxiliares IS SELECT DISTINCT ID_FUNCIONARIO, NOMBRE_FUNCIONARIO FROM (SELECT *
    FROM PROCEDIMIENTO P INNER JOIN FUNCIONARIO F ON
P.ID_AUXILIAR = F.ID_FUNCIONARIO WHERE FUNCIONARIO_ENTREGA = ID_FUN);
  funcionario LONG;
  reg_aux auxiliares%ROWTYPE;
BEGIN
  fecha := SYSDATE;
  SELECT NOMBRE_FUNCIONARIO INTO funcionario FROM FUNCIONARIO WHERE ID_FUNCIONARIO = ID_FUN;
  HTP.p('ACTA DE ENTREGA DE PUESTO DE TRABAJO');
  HTP.p('');
  HTP.p('Fecha de entrega: ' || fecha);
  HTP.p('');
  HTP.p('-----');
  HTP.p(RPAD('ID Activo', 15) || RPAD('REFERENCIA', 25) || RPAD('NOMBRE', 30) || RPAD('MARCA', 17) || RPAD('DETALLES
ADICIONALES', 25));
  HTP.p('-----');
  OPEN activos;
  LOOP

```

```

        FETCH activos INTO reg;

        EXIT WHEN activos%NOTFOUND;

        HTP.p(RPAD(TO_CHAR(reg.ID_ACTIVO), 15) || RPAD(TO_CHAR(reg.REFERENCIA_ACTIVO), 25) ||
RPAD(TO_CHAR(reg.NOMBRE_ACTIVO), 30) ||
RPAD(TO_CHAR(reg.MARCA_ACTIVO), 17) || RPAD(TO_CHAR(reg.MATERIAL_ACTIVO), 17) ||
RPAD(TO_CHAR(reg.COLOR_ACTIVO), 17) || RPAD(TO_CHAR(reg.TAMANO_ACTIVO), 17) || 'Componentes: ' ||
RPAD(TO_CHAR(reg.COMPONENTES_ACTIVO), 17));

        END LOOP;

        CLOSE activos;

        HTP.p('-----');

        HTP.p('Entrega(n): ');

        OPEN auxiliares;

        LOOP

            FETCH auxiliares INTO reg_aux;

            EXIT WHEN auxiliares%NOTFOUND;

            HTP.p(reg_aux.NOMBRE_FUNCIONARIO);

            END LOOP;

            CLOSE auxiliares;

            HTP.p('');

            HTP.p('Recibe: ' || funcionario);

        END;
    
```

Funciones

El procedimiento implementado para generar una factura, se implementó también en su versión de función de PL/SQL. A continuación se presenta.

```

create or replace FUNCTION genFactura(ID_COM IN NUMBER)
RETURN LONG
IS
    CURSOR activos IS SELECT ID_ACTIVO, REFERENCIA_ACTIVO, NOMBRE_ACTIVO, MARCA_ACTIVO, VALOR_ACTIVO FROM ACTIVO WHERE
ID_COMPRA = ID_COM ORDER BY NOMBRE_ACTIVO, ID_ACTIVO;

    total NUMBER;
    suma NUMBER;
    reg activos%ROWTYPE;
    fecha DATE;
    result LONG;

BEGIN
    SELECT VALOR_COMPRA INTO total FROM COMPRA WHERE ID_COMPRA = ID_COM;
    SELECT SUM(VALOR_ACTIVO) INTO suma FROM ACTIVO WHERE ID_COMPRA = ID_COM;
    SELECT FECHA_COMPRA INTO fecha FROM COMPRA WHERE ID_COMPRA = ID_COM;

    result := 'FECHA DE COMPRA: ' || fecha;
    result := result || chr(13)|| chr(10);

```

```

result := result || 'ID FACTURA: ' || ID_COM;
result := result || chr(13)||chr(10);
result := result || '-----';
result := result || chr(13)||chr(10);
result := result || RPAD('ID Activo', 15) || RPAD('REFERENCIA', 25) || RPAD('NOMBRE', 30) || RPAD('MARCA', 17) || 'VALOR';
result := result || '-----';
result := result || chr(13)||chr(10);
OPEN activos;
LOOP
  FETCH activos INTO reg;
  EXIT WHEN activos%NOTFOUND;
  result := result ||RPAD(TO_CHAR(reg.ID_ACTIVO), 15) || RPAD(TO_CHAR(reg.REFERENCIA_ACTIVO), 25) || RPAD(TO_CHAR(reg.NOMBRE_ACTIVO), 30) ||
RPAD(TO_CHAR(reg.MARCA_ACTIVO), 17) || TO_CHAR(reg.VALOR_ACTIVO, '$999,999,999.00') ||
' COP';
  result := result || chr(13)||chr(10);
END LOOP;
result := result ||'-----';
result := result || chr(13)||chr(10);
result := result || 'Suma: ' || TO_CHAR(suma, '$999,999,999.00') || ' COP';
result := result || chr(13)||chr(10);
result := result || '.....';
result := result || chr(13)||chr(10);
result := result || 'Valor Registrado en Factura: ' || TO_CHAR(total, '$999,999,999.00') || ' COP';
result := result || chr(13)||chr(10);
result := result || '.....';
result := result || chr(13)||chr(10);
result := result || 'Desfase: ' || TO_CHAR(total-suma, '$999,999,999.00') || ' COP';
result := result || chr(13)||chr(10);
CLOSE activos;
RETURN result;
END;

```

Se implementó una función para obtener la oficina en la que trabajaría un funcionario evaluando como parámetro el ID_FUNCIONARIO.

```

create or replace FUNCTION getOficinaByFuncionario(id_fun IN NUMBER)
RETURN NUMBER
IS
id_loc NUMBER;

```

```

BEGIN
    SELECT MIN(id_locacion) INTO id_loc FROM (SELECT id_funcionario, id_area FROM funcionario)
        NATURAL JOIN AREA NATURAL JOIN LOCACION WHERE id_fun = id_funcionario;
    RETURN id_loc;
END getOficinaByFuncionario;

```

La función anterior no es útil por si sola, porque la llave primaria de LOCACION es compuesta. Puede haber oficinas de distintos edificios con el mismo identificador. Por eso se requirió implementar la siguiente función.

```

create or replace FUNCTION getEdificioByOficina(id_ofi IN NUMBER)
RETURN NUMBER
IS
id_edi NUMBER;

BEGIN
    SELECT id_edificio INTO id_edi FROM LOCACION WHERE id_ofi = id_locacion;
    RETURN id_edi;
END getEdificioByOficina;

```

De esta manera, para implementar esta función, se debe hacer la siguiente composición:

```
getEdificioByOficina(getOficinaByFuncionario(ID_FUNCIONARIO));
```

Así, al ejecutar ambas funciones, es posible obtener una llave primaria apropiada del edificio y la oficina en la cual trabaja un funcionario administrativo de la universidad.

Adicional a estas dos funciones, se generaron tres más que son útiles para el equipo de Contabilidad de la universidad. Estas funciones permiten obtener la información contable de cada activo.

La primera función, calcula los días transcurridos hasta la fecha, contando desde la fecha en que se compró el activo:

```

create or replace FUNCTION dias_desde_compra(fecha_compra IN DATE)
RETURN NUMBER
IS
dias NUMBER;
BEGIN
    dias := ROUND(SYSDATE - fecha_compra, 0);
    RETURN dias;
END dias_desde_compra;

```

La segunda función, calcula el porcentaje de depreciación transcurrido para un activo según la fecha en que se compró. Usa la función anterior para hacer el cálculo.

```

create or replace FUNCTION deprec_trans(fecha_compra IN DATE, deprec_total IN NUMBER)
RETURN NUMBER
IS
deprec NUMBER;
BEGIN

```

```

deprec := ROUND((dias_desde_compra(fecha_compra) * deprec_total)/365, 2);

RETURN deprec;

END deprec_trans;

```

La última función calcula el valor actual del activo según el índice de depreciación transcurrido hasta la fecha. Usa la función anterior para generar el resultado.

```

create or replace FUNCTION valor_activo_hoy(valor_activo IN NUMBER, deprec_total IN NUMBER, fecha_compra IN DATE)
RETURN NUMBER

IS

valor_hoy NUMBER;

BEGIN

valor_hoy := ROUND(valor_activo-(valor_activo*deprec_trans(fecha_compra, deprec_total))/100, 0) ;

RETURN valor_hoy;

END valor_activo_hoy;

```

Estas funciones permiten ejecutar la consulta 18 mucho con mayor facilidad, al encapsular la complejidad de dichas operaciones. A continuación se muestra un ejemplo:

```

CREATE VIEW depreciacion AS
SELECT ID_ACTIVOS, NOMBRE_ACTIVOS, VALOR_ACTIVOS, FECHA_COMPRA, DEPRECIACION_ACTIVOS,
DIAS_DESDE_COMPRA(FECHA_COMPRA) DIAS_TRANSCURRIDOS,
DEPREC_TRANS(FECHA_COMPRA, DEPRECIACION_ACTIVOS) DEPRECIACION_ACUMULADA,
VALOR_ACTIVO_HOY(VALOR_ACTIVOS, DEPRECIACION_ACTIVOS, FECHA_COMPRA) VALOR_ACTIVO_HOY
FROM ACTIVOS NATURAL JOIN COMPRA

```

Disparadores

Se generaron dos autonumeradores para las tablas PROCEDIMIENTO y ACTIVO, de manera que cada vez que se inserte un registro para una de estas dos tablas, no importa cuál valor se escribe en el campo de la llave primaria, el valor que se almacena en la base de datos para este atributo es el siguiente número de una secuencia, de manera que el usuario no tenga que recordar cual es el último ID registrado para esa tabla.

Para hacer esto en la versión de ORACLE 11g se necesita crear una secuencia por cada autonumerador.

```

CREATE SEQUENCE procedimiento_secuencia;
CREATE SEQUENCE activo_secuencia START WITH 2793;

```

Para la secuencia de activos, se necesita especificar el valor inicial de la secuencia, pues ya habían 2793 activos registrados en la base de datos según el código de inserción generado.

Habiendo realizado esto, se procede a crear los disparadores.

Estos son dos disparadores del tipo BEFORE INSERT a nivel de fila.

```

create or replace TRIGGER autonum_procedimiento
BEFORE INSERT ON procedimiento
FOR EACH ROW
BEGIN
SELECT procedimiento_secuencia.nextval

```

```

INTO :new.id_proce
FROM dual;
END;

create or replace TRIGGER autonum_activos
BEFORE INSERT ON activo
FOR EACH ROW
BEGIN
SELECT activo_secuencia.nextval
INTO :new.id_activo
FROM dual;
END;

```

Se generarán dos disparadores que garantizan la actualización de la tabla ACTIVO cuando se hace una entrega de un activo como puesto de trabajo a un funcionario. El primero asigna el ID_FUNCIONARIO al activo en cuestión, formalizando la entrega del activo del puesto de trabajo.

```

create or replace TRIGGER update_funcionario_on_activo
AFTER INSERT ON procedimiento
FOR EACH ROW
BEGIN
IF (:NEW.tipo_procedimiento = 'E') THEN
    UPDATE ACTIVO
    SET id_funcionario = :NEW.funcionario_entrega
    WHERE id_activo = :NEW.id_activo;
END IF;
END;

```

El segundo disparador se asegura de actualizar la ubicación física del activo haciendo uso de las dos funciones creadas para encontrar la oficina y el edificio en el que trabajaría un funcionario.

```

create or replace TRIGGER update_locacion_on_activo
AFTER INSERT ON procedimiento
FOR EACH ROW
BEGIN
IF (:NEW.tipo_procedimiento = 'E') THEN
    UPDATE ACTIVO
    SET id_locacion = getOficinaByFuncionario(:NEW.funcionario_entrega),
        id_edificio = getEdificioByOficina(getOficinaByFuncionario(:NEW.funcionario_entrega))
    WHERE id_activo = :NEW.id_activo;
END IF;
END;

```

Otro disparador útil para la implementación de la base de datos es el siguiente, en el que se retira el valor ID_FUNCIONARIO de un activo que es devuelto por un funcionario al retirarse de la compañía.

```
create or replace TRIGGER remove_funcionario_on_activo
```

```

AFTER INSERT ON procedimiento
FOR EACH ROW
BEGIN
  IF (:NEW.tipo_procedimiento = 'P') THEN
    UPDATE ACTIVO
    SET id_funcionario = NULL
    WHERE id_activo = :NEW.id_activo;
  END IF;
END;

```

Un último disparador busca cambiar el estado de revisión de un activo de la tabla PROCEDIMIENTO cuando ya ha sido reparado por un auxiliar de mantenimiento, de manera que se cambie dicho estado 'M' a NULL que de acuerdo con el diseño de dicha estructura, significaría que el activo está en buenas condiciones.

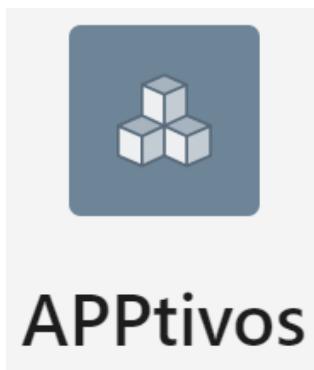
```

create or replace TRIGGER regReparacion
BEFORE INSERT ON procedimiento
FOR EACH ROW
BEGIN
  IF (:NEW.tipo_procedimiento = 'M') THEN
    UPDATE PROCEDIMIENTO
    SET ESTADO_REVISION = NULL
    WHERE id_activo = :NEW.id_activo;
  END IF;
END;

```

DESARROLLO DE LA APLICACIÓN

El nombre de la aplicación es APPtivos.



La aplicación se compone de 14 páginas.

Cada página está diseñada para una entidad interesada específica.

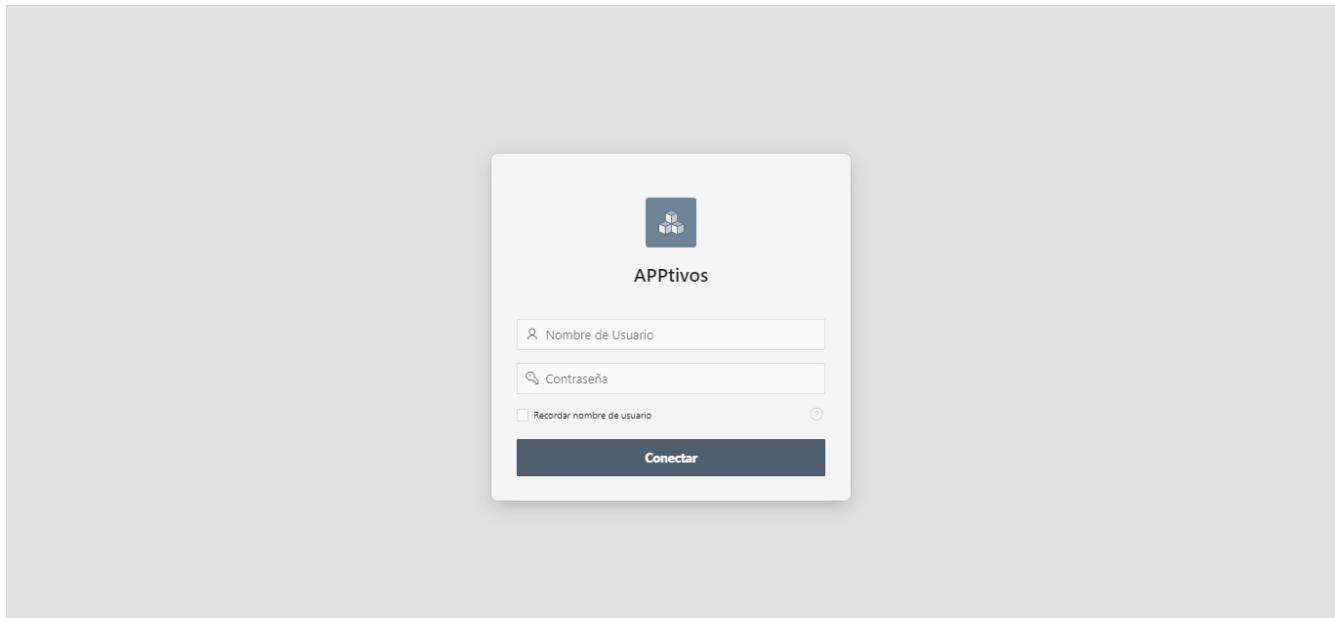
Es posible garantizar accesos específicos para cada tipo de usuario, de manera que cada uno solo vea las páginas a las que tiene acceso. Esto se hace creando roles o perfiles de usuario y especificando en la sección de Seguridad de la página el tipo de usuario que puede acceder a ella. Para esta versión, no se implementó, con el fin de recorrer

las páginas con mayor facilidad sin tener que desconectarse y conectarse cada vez a la aplicación, cada vez que se quiera ver un cambio entre las tablas.

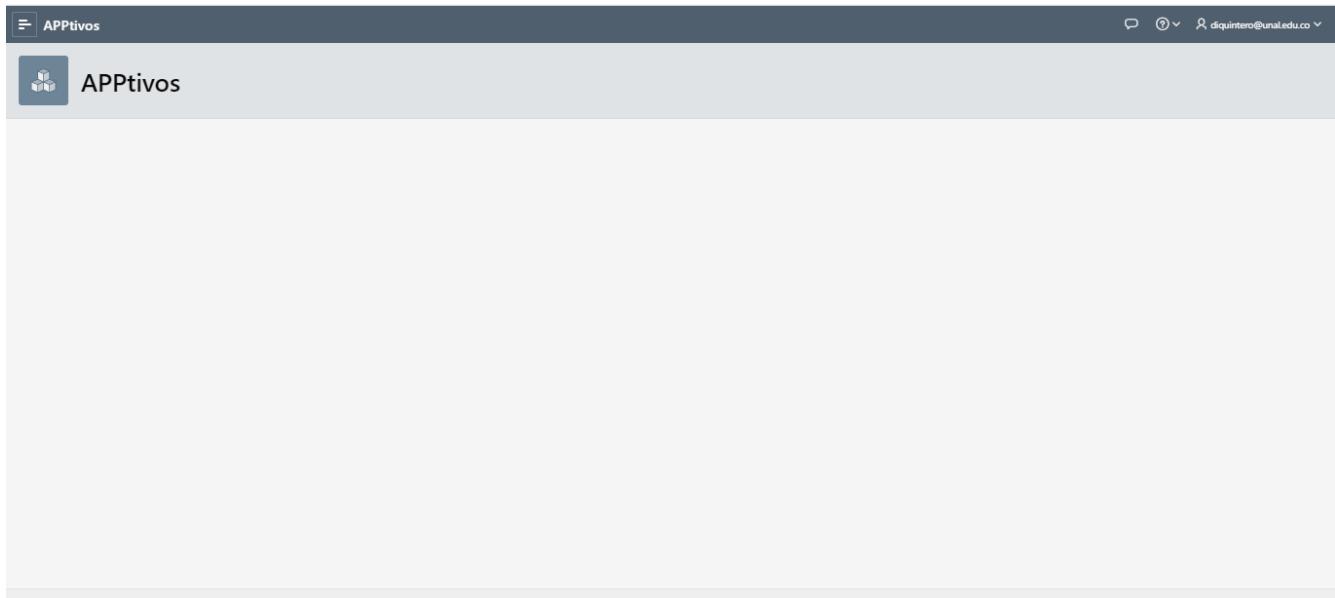
Los tipos de usuarios de la aplicación son:

- Auxiliares de Inventarios y personal de Compras y Almacén
- Auxiliares de Mantenimiento
- Personal administrativo de Logística, especialmente del área de Servicios Generales
- Personal de Contabilidad y Finanzas
- Personal de Gestión Humana y Desarrollo Organizacional

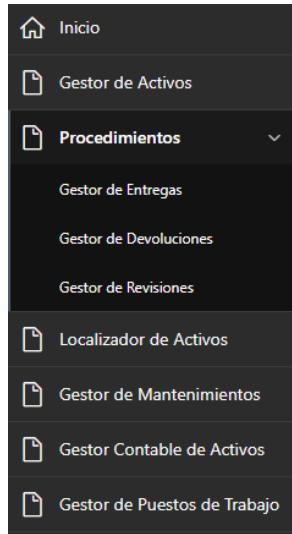
Los usuarios deben autenticarse en la siguiente página de Conexión, una vez abran la aplicación:



Una vez ingresen, la primera página de la aplicación es la de Inicio:



En la parte izquierda de la aplicación hay un menú de navegación por la aplicación que incluye una lista de enlaces a las páginas principales de la misma:



A continuación se indican los tipos de usuario que podrían acceder a cada una de estas páginas:

Página	Tipo de Usuario
Gestor de Activos	
Procedimientos	Auxiliares de Inventarios y personal de Compras y Almacén
Gestor de Entregas	
Gestor de Devoluciones	
Gestor de Revisiones	
Localizador de Activos (Gestor Logístico de Activos)	Personal administrativo de Logística, especialmente del área de Servicios Generales
Gestor de Mantenimientos	Auxiliares de Mantenimiento
Gestor Contable de Activos	Personal de Contabilidad y Finanzas
Gestor de Puestos de Trabajo	Personal de Gestión Humana y Desarrollo Organizacional

Ahora, se describirá cada una de estas páginas junto con algunos detalles de su diseño.

Gestor de Activos

Esta página tiene como objetivo mostrar todo el esquema de la tabla ACTIVO, de manera que pueda ser consultado por los auxiliares de inventarios y el personal de compras y almacén.

La página se compone de dos regiones. Un informe basado en un formulario que permite consultar la tabla ACTIVO y hacer modificaciones de registros en dicha tabla.

Gestor de Activos

INSERTAR ACTIVO

ID Compra

Referencia

Categoría (?)

Nombre

Marca

Valor

		Q ▾		Ir		Acciones ▾									
		ID Activo	ID Compra	Referencia Activo	Categoría Activo	Nombre Activo	Marca Activo	Valor Activo	Depreciación Activo	Id Funcionario	Id Edificio	Id Locación	Material Activo	Color Activo	
	1361	2020015	R9RKI495J	SGT	Escrítorio	Lexington	266060	8	1018005313	10	53002	Metálico	B		
	1362	2020015	GR5QRVC329	SGT	Silla Giratoria	Ibaby	164145	8	1018005313	10	53002	Cuero	N		
	1363	2020018	XE8IBEYEFPDF6FWZS6K5	DTS	Computador de Escritorio	Toshiba	2730920	10	1018005314	10	53002				
	1364	2020018	Z28WNNPA0	SGT	Escrítorio	Lexington	289636	7	1018005314	10	53002	Madera	B		
	1293	2020044	K3UBUZ3LT6	SGT	Silla Giratoria	Ibaby	255410	8	1018005326	3	55001	Sintético	G		
	1294	2020022	P6VTAVS4ZSQJ9G1MBANP	DTS	Computador de Escritorio	Acer	2779655	10	1018005065	9	11001				
	1295	2020022	TH9PV47MJ	SGT	Escrítorio	Flexa	297382	7	1018005065	9	11001	Metálico	M		
	1296	2020022	OTITX19020	SGT	Silla	CashOffice	209791	8	1018005065	9	11001	Malla	N		

La segunda región es un formulario creado con una serie de campos de texto, campos de autocompletado, listas de selección y un botón para insertar un registro nuevo en la estructura. La siguiente imagen muestra algunos de estos campos y el botón de INSERTAR al final.

Edificio

Locación

Material

Color

Tamaño

Componentes

Ingrese cada componente separado por una coma y un espacio en blanco

INSERTAR

Algunos de estos campos del formulario cuentan con una consulta SQL asociada que permite desplegar opciones al usuario. A continuación se presentan estas consultas:

ATRIBUTO	CAMPO DE TEXTO	CONSULTA SQL
CATEGORIA_ACTIVO	AC_CAT	SELECT DISTINCT CATEGORIA_ACTIVO AS display_value, CATEGORIA_ACTIVO AS return_value FROM ACTIVO ORDER BY display_value
NOMBRE_ACTIVO	AC_NOM	SELECT DISTINCT NOMBRE_ACTIVO AS display_value, NOMBRE_ACTIVO AS return_value FROM ACTIVO ORDER BY display_value
MARCA_ACTIVO	AC_MAR	SELECT DISTINCT MARCA_ACTIVO AS display_value, MARCA_ACTIVO AS return_value FROM ACTIVO ORDER BY display_value

ID_EDIFICIO	ACEDI	SELECT DISTINCT (NOMBRE_SEDE ' - ' NOMBRE_EDIFICIO) AS display_value, ID EDIFICIO AS return_value FROM EDIFICIO NATURAL JOIN SEDE ORDER BY display_value
ID_LOCACION	ACLOC	SELECT DISTINCT NOMBRE_LOCACION AS display_value, ID LOCACION AS return_value FROM LOCACION ORDER BY display_value
MATERIAL_ACTIVO	ACMAT	SELECT DISTINCT MATERIAL_ACTIVO AS display_value, MATERIAL_ACTIVO AS return_value FROM ACTIVO ORDER BY display_value
COLOR_ACTIVO	ACCOL	SELECT DISTINCT COLOR_ACTIVO AS display_value, COLOR_ACTIVO AS return_value FROM ACTIVO ORDER BY display_value
TAMANO_ACTIVO	ACTAM	SELECT DISTINCT TAMANO_ACTIVO AS display_value, TAMANO_ACTIVO AS return_value FROM ACTIVO ORDER BY display_value
COMPONENTES_ACTIVO	ACCOM	SELECT DISTINCT COMPONENTES_ACTIVO AS display_value, COMPONENTES_ACTIVO AS return_value FROM ACTIVO ORDER BY display_value

El botón de inserción para este formulario tiene la siguiente sentencia SQL asociada:

```
INSERT INTO ACTIVO VALUES (9999, TO_NUMBER(:AC_COMPR), :AC_REF, :AC_CAT, :AC_NOM, :AC_MAR, TO_NUMBER(:AC_VAL), TO_NUMBER(:AC_DEP), TO_NUMBER(:AC_FUN), :ACEDI, TO_NUMBER(:ACLOC), :ACMAT, :ACCOL, :ACTAM, :ACCOM);
```

Edición de Activos

Gestor de Activos \

Edición de Activos

Edición de Activos

Id Activo 1361	Id Compra 2020015	Referencia Activo R9RKI495J	Categoría Activo SGT
Nombre Activo Escritorio	Marca Activo Lexington	Valor Activo 266060	Depreciación Activo 8
Id Funcionario 1018005313	Id Edificio Sede Principal - Bloque J	Id Locación 53002	Material Activo Metálico
Color Activo Blanco	Tamaño Activo 1.20 m x 0.80 m		Componentes Activo
<input type="button" value="Cancelar"/>		<input type="button" value="Aplicar Cambios"/>	

La tercera página de la aplicación es el formulario que se genera cuando un usuario hace clic sobre el botón de editar registro en la tabla de ACTIVO.

Esta página se genera de forma automática por el creador de páginas de APEX. Simplemente se configuraron los campos de texto según las consultas listadas anteriormente.

También, se eliminó el botón de BORRAR para evitar que el personal de esta área elimine activos, lo cual solo le corresponde al personal de Contabilidad.

Asimismo se eliminó el botón de CREAR, puesto que ya se había creado el formulario de inserción en la página anterior Gestor de Activos.

Procedimientos

El objetivo de esta página es brindar información relevante al área de Compras y Almacén.

Se compone de varias regiones basadas en informes interactivos generados por consultas y una gráfica basada en una tabla.

La primera región muestra una tabla con los activos que después de haber sido revisados por los auxiliares de inventarios y/o de mantenimiento, necesitan ser reemplazados por unos nuevos, convirtiéndose así en solicitudes de compra.

The screenshot shows a web-based application interface titled "Procedimientos". Below it is a table titled "Solicitudes de Compra" (Purchase Requests). The table has the following columns:

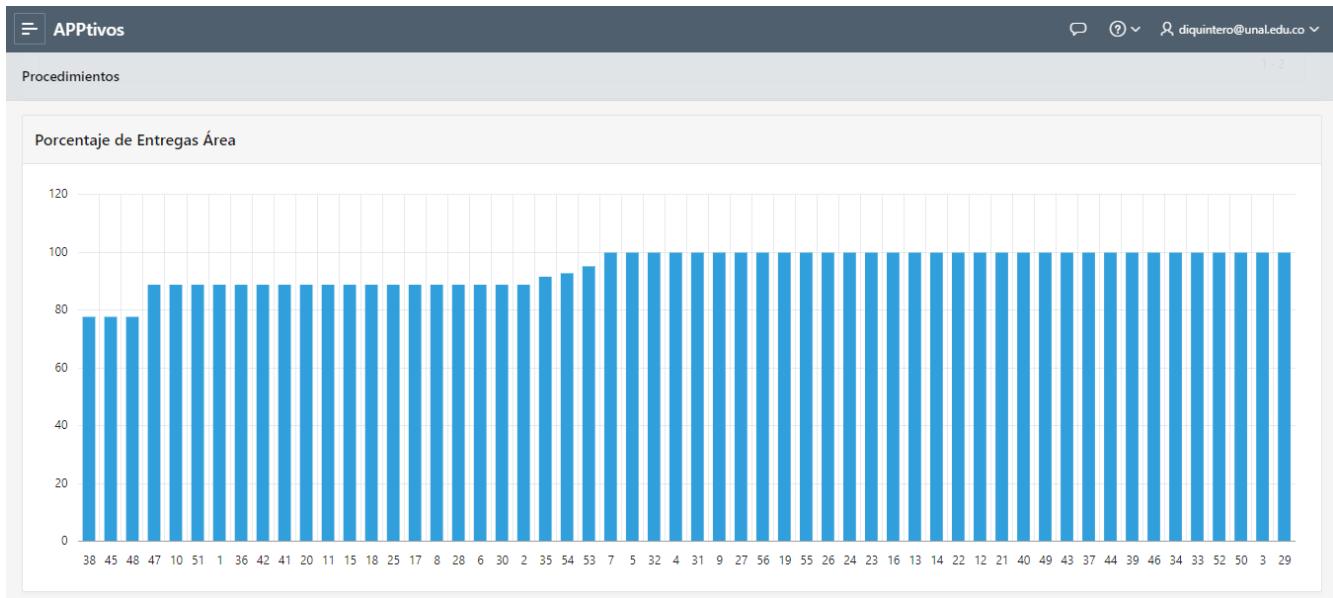
Solicitudes de Compra															
Q. v				Ir	Acciones v										
ID Activo	Referencia Activo	ID Compra	Fecha Compra	Nombre Activo	Marca Activo	Observaciones	ID Auxiliar	Fecha Procedimiento	Material Activo	Color Activo	Tamaño Activo	Componentes Activo	Valor Compra	D	
2793	5W2YJ7U9AG	2020032	08/01/2020	Silla Giratoria	EBS	Después de consultar con el personal de Servicios Generales, se determinó que no vale la pena hacer una reparación.	1019004003	21/07/2020	Sintético	Azul Oscuro			\$331,483.00 COP	\$3 CC	
1054	UTKTGB23JHEKO0WB0TDU15YLM	2020016	25/05/2020	Equipo de Sonido	Woltu		1019004003	22/07/2020			Juego de Microfonos, Soportes, Amplificador, Mezclador, Altavoces		\$4,406,958.00 COP	\$4 CC	

La tabla se generó a partir de la siguiente consulta SQL

```

SELECT ID_ACTIVO, REFERENCIA_ACTIVO, ID_COMPRA, FECHA_COMPRA, NOMBRE_ACTIVO, MARCA_ACTIVO,
(TO_CHAR(VALOR_ACTIVO,'$999,999,999.99') || ' COP') VALOR_COMPRA,
(TO_CHAR(VALOR_ACTIVO_HOY(VALOR_ACTIVO, DEPRECIACION_ACTIVO, FECHA_COMPRA), '$999,999,999.99') || ' COP')
VALOR_DEPRECIADO,
OBSERVACIONES, ID_AUXILIAR, FECHA_PROCEDIMIENTO, MATERIAL_ACTIVO, COLOR_ACTIVO, TAMAÑO_ACTIVO, COMPONENTES_ACTIVO
FROM ACTIVO NATURAL JOIN COMPRA NATURAL JOIN PROCEDIMIENTO
WHERE ESTADO_REVISION = 'R'
```

Luego se presenta una gráfica basada en una tabla que también se encuentra en esta página de la aplicación:



Busca mostrar el porcentaje de puestos asignados según la cantidad de funcionarios que trabajan en un área específica. La universidad tiene 56 áreas, y por ende hay 56 columnas.

Precisamente, la tabla que se relaciona con esta gráfica, es la siguiente:

Avance en Puestos Entregados Por Área				
Id Area	Nombre Area	Puestos Asignados	Empleados Existentes	Porcentaje Asignado
38	Ayudas Educativas	7	9	77,78
45	Servicios del Medio Universitario	7	9	77,78
48	Jefatura de Derecho	7	9	77,78
36	Infraestructura	8	9	88,89
51	Centros de Conciliación	8	9	88,89
10	Programa Sembrar Paz	8	9	88,89
47	Servicios del Medio Universitario	8	9	88,89
20	Permanencia Estudiantil	8	9	88,89
11	Articulación con la Educación Media	8	9	88,89
15	Sistemas Integrados de Gestión	8	9	88,89

Ambos objetos se generaron a partir de la siguiente consulta SQL:

```
WITH puestos_asignados AS
  (SELECT ID_AREA, COUNT(ID_AREA) PUESTOS_ASIGNADOS
   FROM (SELECT ID_FUNCIONARIO FROM ACTIVO WHERE ID_FUNCIONARIO IS NOT NULL GROUP BY ID_FUNCIONARIO)
   NATURAL JOIN FUNCIONARIO WHERE ID_AREA IS NOT NULL GROUP BY ID_AREA),
  empleados_por_cargo AS
```

```

(SELECT ID_AREA, COUNT(*) EMPLEADOS_EXISTENTES
FROM FUNCIONARIO WHERE ID_AREA IS NOT NULL GROUP BY ID_AREA)

SELECT puestos_asignados.ID_AREA,
       AREA.NOMBRE_AREA,
       PUESTOS_ASIGNADOS,
       EMPLEADOS_EXISTENTES,
       ROUND((PUESTOS_ASIGNADOS*100)/EMPLEADOS_EXISTENTES, 2) PORCENTAJE_ASIGNADO
FROM (puestos_asignados INNER JOIN empleados_por_cargo
      ON puestos_asignados.ID_AREA = empleados_por_cargo.ID_AREA) INNER JOIN AREA ON puestos_asignados.ID_AREA = AREA.ID_AREA
ORDER BY PORCENTAJE_ASIGNADO

```

La página también cuenta con una tabla que muestra las compras realizadas hasta el momento y que han sido registradas en la base de datos. Por medio de una proyección generalizada, se especifican ciertos formatos para los resultados:

Compras Realizadas				
Cod Compra	Año Compra	Mes Compra	Dia Compra	Valor
0	2020	Mayo	07	\$31,798,317.00 COP
1	2020	Abril	12	\$141,805,710.00 COP
2	2020	Junio	18	\$48,520,628.00 COP
3	2020	Abril	05	\$86,980,952.00 COP
4	2020	Febrero	01	\$81,294,464.00 COP
5	2020	Enero	25	\$46,976,131.00 COP
6	2020	Mayo	19	\$43,955,033.00 COP
7	2020	Mayo	18	\$75,535,808.00 COP
8	2020	Marzo	19	\$57,046,177.00 COP
9	2020	Marzo	21	\$87,331,551.00 COP
10	2020	Febrero	26	\$96,479,091.00 COP
11	2020	Enero	08	\$94,708,209.00 COP
12	2020	Enero	21	\$49,017,730.00 COP

La consulta SQL que genera este informe interactivo, es la siguiente:

```

SELECT TO_NUMBER(SUBSTR(TO_CHAR(id_compra),5, 3)) cod_compra,
       TO_CHAR(fecha_compra, 'yyyy') año_compra,
       TO_CHAR(fecha_compra, 'Month') mes_compra,
       TO_CHAR(fecha_compra, 'dd') dia_compra,
       (TO_CHAR(VALOR_COMPRA,'$999,999,999.99') || ' COP') VALOR

```

FROM COMPRA

Otro informe interactivo que se decidió incluir es el de una tabla en la que se muestren las compras cuyos montos no coinciden con la suma total de los activos asociados a dicha compra.

Irregularidades en Compras				
La siguiente tabla muestra las inconsistencias que puedan existir entre el valor total de los activos que pertenecen a cierta compra y el valor total de la compra. Un valor negativo significa que la suma total de los activos registrados con una factura supera el monto registrado en la factura.				
IdCompra	Suma Activos	ValorCompra	Desfase	
2020032	\$53,817,249.00 COP	\$53,017,249.00 COP	-\$800,000.00 COP	
2020033	\$30,699,963.00 COP	\$31,199,963.00 COP	\$500,000.00 COP	
2020041	\$34,148,577.00 COP	\$33,903,577.00 COP	-\$245,000.00 COP	

La consulta asociada a este objeto es:

```
WITH sumas AS (SELECT ID_COMPRA, SUM(VALOR_ACTIVO) SUMA FROM ACTIVO GROUP BY ID_COMPRA)

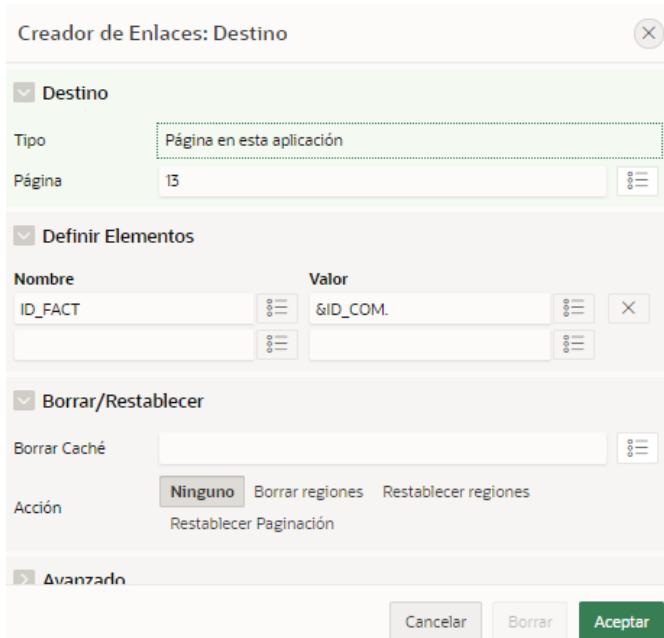
SELECT sumas.ID_COMPRA, (TO_CHAR(SUMA, '$999,999,999.00') || ' COP') SUMA_ACTIVOS, (TO_CHAR(COM.VALOR_COMPRA,
'$999,999,999.00') || ' COP') VALOR_COMPRA, ((SELECT VALOR_COMPRA FROM COMPRA C WHERE C.ID_COMPRA =
sumas.ID_COMPRA)-SUMA), ('$999,999,999.00') || ' COP') DESFASE

FROM sumas INNER JOIN COMPRA COM ON sumas.ID_COMPRA = COM.ID_COMPRA

WHERE SUMA < (SELECT VALOR_COMPRA FROM COMPRA WHERE ID_COMPRA = sumas.ID_COMPRA) OR SUMA > (SELECT VALOR_COMPRA FROM
COMPRA WHERE ID_COMPRA = sumas.ID_COMPRA)
```

Se incluyó además un componente llamado *Generador de Facturas* compuesto por un campo de texto llamado ID_COM que recibe como parámetro el ID de la factura a consultar. El usuario debe escribir el número de la factura en dicho campo, presionar ENTER y luego hacer clic sobre el botón GENERAR.

Este botón tiene un comportamiento asociado dado por las siguientes especificaciones:



Donde 13 es el número de la página en la cual se presentará el resultado de la operación. Y el elemento ID:FACT es un área de texto encontrado en dicha página, como se ve a continuación:

The screenshot shows a user interface for a facture application. At the top, there's a navigation bar with the logo 'APPtivos' and a menu icon. Below it, a breadcrumb trail says 'Procedimientos \ Factura'. The main content area has a form with a single input field. The field is labeled 'ID Factura:' and contains the value 'Nuevo 2020000', which is highlighted with a dashed border.

El valor que se registró en ID_COM es enviado al elemento ID_FACT como una variable que será utilizada por una región de contenido PL/SQL dinámico llamada Detalle. El código PL/SQL de esta región es el siguiente:

```
HTP.p ('<a download="factura.txt" id="downloadlink" ><b>factura.txt</b></a> <br> <br> <br>');
DECLARE
    nulo EXCEPTION;
BEGIN
    HTP.p ('<span id="TEST_ID">');
    IF :ID_FACT IS NULL THEN
        RAISE nulo;
    END IF;
    HTP.p (genFactura(TO_NUMBER(:ID_FACT)));
    HTP.p ('</span>');
    EXCEPTION
    WHEN nulo THEN
        RETURN;
END;
```

Se utilizan funciones del paquete HTP de HTML para conseguir imprimir texto en la renderización de la página web. Y se hace uso de la función definida como genFactura(ID_FACTURA) para inscribir todo el texto en un documento HTML delimitado por las etiquetas ` `.

La primera línea de código genera un link de descarga con los atributos presentados en la etiqueta de apertura `<a>`. Para garantizar que el link funciona, se hace uso de un código en JavaScript facilitado por [2] y que se presenta a continuación:

```
(function () {
    var textFile = null,
        makeTextFile = function (text) {
            var data = new Blob([text], {type: 'text/plain'});
            if (textFile !== null) {
                window.URL.revokeObjectURL(textFile);
            }
            textFile = URL.createObjectURL(data);
        }
    return {
        download: function (text) {
            makeTextFile(text);
            var a = document.createElement('a');
            a.href = textFile;
            a.download = 'factura.txt';
            a.click();
        }
    };
})()
```

```

textFile = window.URL.createObjectURL(data);

return textFile;};

var create = document.getElementById('create'),
textbox = document.getElementById('TEST_ID');

var link = document.getElementById('downloadlink');

link.href = makeTextFile(textbox.innerHTML.replace(/&/g,"&").trim());
link.style.display = 'block';

})();

```

Facilitado por Muhammad Abdul Qaium en su blog Qaium's IT Demonstration

El código en JavaScript accede al documento HTML y el enlace de descarga por su identificador, según se definió en el campo de contenido dinámico PL/SQL después de haber definido las especificaciones del archivo a descargar (en este caso un documento de texto plano .txt). Finalmente, introduce el texto generado por la función en dicho archivo.

Lo anterior fue igualmente implementado para generar un acta de entrega usando el procedimiento definido para ello. La implementación es prácticamente igual, salvo que se hace desde la página *Gestor de Entregas* hacia la página 14 *Acta de Entrega*, cambiando los valores correspondientes.

The screenshot shows a web-based application interface. On the left, there is a vertical navigation menu with items like 'Inicio', 'Gestor de Activos', 'Procedimientos' (with sub-options 'Gestor de Entregas', 'Gestor de Devoluciones', 'Gestor de Revisiones'), 'Localizador de Activos', 'Gestor de Mantenimientos', 'Gestor Contable de Activos', 'Gestor de Puestos de Trabajo', and 'Administración'. The main content area has a header 'Procedimientos \ Gestor de Entregas'. Below the header is a table with columns: Id Proce, Id Activo, Funcionario Entrega, Fecha Procedimiento, Observaciones, Id Auxiliar, Nombre Activo, Nombre Funcionario, Id Area, Nombre Area, and Nombre Auxiliar. The table contains five rows of data. At the bottom of the main content area, there is a form titled 'Generador de Actas' with a text input field containing 'Id Fun 1018006078', a note saying 'Presiona ENTER una vez ingreses el ID de la factura. Luego haz CLIC en el botón GENERAR.', and a 'Generar' button. At the very bottom of the page, there is a footer bar with links: 'Inicio', 'Aplicación 11918', 'Editar Página 5', 'Sesión', 'Ver Depuración', 'Depurar', 'Información de página', 'Edición Rápida', 'Acumulador de Temas', and a help icon.

Id Proce	Id Activo	Funcionario Entrega	Fecha Procedimiento	Observaciones	Id Auxiliar	Nombre Activo	Nombre Funcionario	Id Area	Nombre Area	Nombre Auxiliar
222	1564	1018005232	23/07/2020		1019004002	Computador de Escritorio	Edison Bernardo Prieto Molina	39	Soporte Técnico y Telecomunicaciones	Yelson Snader Martínez Soto
242	606	1018005233	23/07/2020		1019004001	Pupitre	Jéssica Alejandra Núñez Valencia	39	Soporte Técnico y Telecomunicaciones	Aura Milena Rodríguez Díaz
223	1492	1018006078	23/07/2020		1019004003	Computador de Escritorio	Carolina Isabel Galindo Álvarez	40	Soporte Sistema Institucional	Diana Marjorie Pineda Vanegas
243	614	1018006079	23/07/2020		1019004002	Televisor	Armando Julián Barrios Montero	40	Soporte Sistema Institucional	Yelson Snader Martínez Soto

APPtivos

- Inicio
- Gestor de Activos
- Procedimientos >
- Localizador de Activos
- Gestor de Mantenimientos
- Gestor Contable de Activos
- Gestor de Puestos de Trabajo
- Administración

Procedimientos \ Gestor de Entregas \

Acta de Entrega

ID Funcionario:

Id Func
1018006078

Detalle Acta:

acta.txt

ACTA DE ENTREGA DE PUESTO DE TRABAJO Fecha de entrega: 24/07/2020 -----
----- ID Activo REFERENCIA NOMBRE MARCA DETALLES ADICIONALES -----
----- 1492 HAPPLE91UXQ32N5J909C Computador de Escritorio Lenovo 21 pulgadas Componentes:
2050 3XQU0NRT4UV68Z3J73Z2 Computador de Escritorio Dell 22 pulgadas Componentes: 2164 190CW5Q907W0V47LMKWR Computador de Escritorio Acer 21 pulgadas Componentes: 2165 UZXYMOSK3 Escritorio Cecotti Madera Marrón 1.20 m x 0.80 m Componentes: 2166 1AHULGW53Y Silla Giratoria SongMics Cuero Azul Oscuro Componentes: -----
----- Entrega(n): Diana Marjorie Pineda Vanegas Yeison Snader Martinez Soto Recibe: Carolina Isabel Galindo Álvarez

Versión 1.0

Inicio Aplicación 11918 Editar Página 14 Sesión Ver Depuración Depurar Información de página Edición Rápida Acumulador de Temas

APPtivos

- Inicio
- Gestor de Activos Archivo Edición Formato Ver Ayuda
- Procedimientos > Fecha de entrega: 24/07/2020
- Localizador de Activos
- Gestor de Mantenimientos
- Administración

acta (3).txt: Bloc de notas

ACTA DE ENTREGA DE PUESTO DE TRABAJO

ID Activo	REFERENCIA	NOMBRE	MARCA	DETALLES ADICIONALES
1492	HAPPLE91UXQ32N5J909C	Computador de Escritorio	Lenovo	21 pulgadas Componentes:
2050	3XQU0NRT4UV68Z3J73Z2	Computador de Escritorio	Dell	22 pulgadas Componentes:
2164	190CW5Q907W0V47LMKWR	Computador de Escritorio	Acer	21 pulgadas Componentes:
2165	UZXYMOSK3	Escritorio	Cecotti	Madera Marrón
2166	1AHULGW53Y Componentes: 1.20 m x 0.80 m	Silla Giratoria	SongMics	Azul Oscuro

Entrega(n):
Diana Marjorie Pineda Vanegas
Yeison Snader Martinez Soto

Recibe: Carolina Isabel Galindo Álvarez

Versión 1.0

Inicio Aplicación 11918 Editar Página 14 Sesión Ver Depuración Depurar Información de página Edición Rápida Acumulador de Temas

Las siguientes páginas son específicas para los procedimientos que deben cumplir los auxiliares de inventarios. Se basan en la tabla PROCEDIMIENTO de la base de datos y su raíz en la aplicación es la página PROCEDIMIENTOS, según se puede ver en el menú desplegable.

Gestor de Entregas

Se generó un formulario de cuatro campos que deben ingresar los auxiliares cuando van a realizar un proceso de entrega de puesto de trabajo a un funcionario.

CAMPO	OBJETO	CONSULTA ASOCIADA
ID_AUXILIAR	PRO_AUX	SELECT DISTINCT (NOMBRE_FUNCIONARIO ' - ID: ' ID_FUNCIONARIO) display_value, ID_FUNCIONARIO return_value FROM FUNCIONARIO WHERE TIPO_AUXILIAR = 'INV'

El botón ENTREGAR tiene asociada la siguiente sentencia SQL:

```
INSERT INTO PROCEDIMIENTO VALUES (9999, TO_NUMBER(:PRO_ACT), TO_NUMBER(:PRO_AUX), 'E', SYSDATE, null, :PRO_OBS,  
TO_NUMBER(:PRO_FUN));
```

El informe interactivo tiene asociada la siguiente consulta SQL:

```
WITH R AS (SELECT ID_PROCE, ID_ACTIVO, NOMBRE_ACTIVO, FUNCIONARIO_ENTREGA, NOMBRE_FUNCIONARIO, ID_AREA, NOMBRE_AREA,  
FECHA_PROCEDIMIENTO, OBSERVACIONES FROM  
  
(SELECT ID_PROCE, ID_ACTIVO, FUNCIONARIO_ENTREGA, OBSERVACIONES, FECHA_PROCEDIMIENTO, TIPO_PROCEDIMIENTO FROM  
PROCEDIMIENTO WHERE TIPO_PROCEDIMIENTO = 'E') NATURAL JOIN  
  
(SELECT ID_ACTIVO, NOMBRE_ACTIVO, ID_FUNCIONARIO FROM ACTIVO) NATURAL JOIN  
  
(SELECT ID_FUNCIONARIO, NOMBRE_FUNCIONARIO, ID_AREA FROM FUNCIONARIO) NATURAL JOIN  
  
(SELECT ID_AREA, NOMBRE_AREA FROM AREA))  
  
SELECT R.*, ID_AUXILIAR, FUNCIONARIO.NOMBRE_FUNCIONARIO NOMBRE_AUXILIAR FROM (R INNER JOIN PROCEDIMIENTO ON R.ID_PROCE  
= PROCEDIMIENTO.ID_PROCE) INNER JOIN FUNCIONARIO ON ID_AUXILIAR = ID_FUNCIONARIO
```

Gestor de Devoluciones

La página sirve como registro de las devoluciones de puestos de trabajo que han hecho los funcionarios al retirarse de la compañía.

APPtivos

- Inicio
- Gestor de Activos
- Procedimientos
 - Gestor de Entregas
 - Gestor de Devoluciones**
 - Gestor de Revisiones
- Localizador de Activos
- Gestor de Mantenimientos
- Gestor Contable de Activos
- Gestor de Puestos de Trabajo
- Administración

Gestor de Devoluciones

INSERTAR DEVOLUCIÓN

ID Activo	ID Auxiliar	Estado del Activo Recibido
Observaciones		En Buenas Condiciones Necesita Comprarse Uno Nuevo Necesita Mantenimiento
<input type="button" value="RECIBIR"/>		

Id Proce	Id Activo	Fecha Procedimiento	Observaciones	Id Auxiliar	Nombre Auxiliar	Estado Activo	Id Funcionario
83	2794	21/07/2020	Hay un problema con el teclado	1019004002	Yelson Snader Martínez Soto	M	1019004002
63	2794	21/07/2020	Hay un problema con el teclado	1019004002	Yelson Snader Martínez Soto	M	1019004002
82	2794	21/07/2020	Hay un problema con el teclado	1019004002	Yelson Snader Martínez Soto	M	1019004002

1 - 3

La estructura es muy similar a la página anterior. El campo ID_AUXILIAR asociado a la lista de selección DEV_AUX cuenta con la misma consulta ya mostrada para el objeto PRO_AUX.

El campo ESTADO_REVISION se configuró así:

Valores Estáticos

Valores

Valor de Visualización	Valor de Retorno
Necesita Mantenimiento	M
Necesita Comprarse Uno Nuevo	R

Ordenar

Y la siguiente especificación:

Estado del Activo Revisado

Deja este espacio en blanco si la reparación fue exitosa

El objeto asociado a ESTADO_REVISION cuenta con la siguiente configuración:

Valores Estáticos

Valores

Valor de Visualización	Valor de Retorno
En Buenas Condiciones	NULL
Necesita Mantenimiento	M
Necesita Comprarse Uno Nuevo	R

Ordenar

La sentencia asociada al botón RECIBIR es:

```
INSERT INTO PROCEDIMIENTO VALUES (9999, TO_NUMBER(:DEV_ACT), TO_NUMBER(:DEV_AUX), 'P', SYSDATE, :DEV_EST, :DEV_OBS, null);
```

El informe interactivo se genera a partir de la siguiente consulta:

```
SELECT ID_PROCE, ID_ACTIVO, FECHA_PROCEDIMIENTO, ESTADO_ACTIVO, OBSERVACIONES, ID_AUXILIAR, ID_FUNCIONARIO, NOMBRE_AUXILIAR
FROM (SELECT ID_PROCE, ID_ACTIVO, FECHA_PROCEDIMIENTO, ESTADO_REVISION ESTADO_ACTIVO, OBSERVACIONES, ID_AUXILIAR, TIPO_PROCEDIMIENTO FROM PROCEDIMIENTO WHERE TIPO_PROCEDIMIENTO = 'P') INNER JOIN
(SELECT ID_FUNCIONARIO, NOMBRE_FUNCIONARIO NOMBRE_AUXILIAR FROM FUNCIONARIO) ON ID_AUXILIAR = ID_FUNCIONARIO
```

Gestor de Revisiones

La séptima página en la aplicación guarda la misma estructura que la anterior salvo por el hecho de que las consultas asociadas al botón REGISTRAR y al informe interactivo son ligeramente distintas:

Id_Proce	Id_Activo	Observaciones	Id_Auxiliar	Nombre_Auxiliar	Estado_Activo	Id_Funcionario	Fecha_Revision
84	2794	Hay un problema con el teclado	1019004002	Yelson Snader Martínez Soto	M	1019004002	21/07/2020
85	2793	Después de consultar con el personal de Servicios Generales, se determinó que no vale la pena hacer una reparación.	1019004003	Diana Marjorie Pineda Vanegas	R	1019004003	21/07/2020
122	1054		1019004003	Diana Marjorie Pineda Vanegas	R	1019004003	22/07/2020
102	785	Fallas en el sistema operativo	1019004002	Yelson Snader Martínez Soto	M	1019004002	22/07/2020

El campo ESTADO_REVISION se configuró así:

Valores Estáticos

Valor de Visualización	Valor de Retorno
Necesita Mantenimiento	M
Necesita Comprarse Uno Nuevo	R

Ordenar

Aceptar

Y la siguiente especificación:

Estado del Activo Revisado

Deja este espacio en blanco si la reparación fue exitosa

Para el botón REGISTRAR la sentencia es:

```
INSERT INTO PROCEDIMIENTO VALUES (9999, TO_NUMBER(:REV_ACT), TO_NUMBER(:REV_AUX), 'R', SYSDATE, :REV_EST, :REV_OBS,
null);
```

Para el informe interactivo, la consulta es:

```
SELECT ID_PROCE, ID_ACTIVO, FECHA_REVISION, ESTADO_ACTIVO, OBSERVACIONES, ID_AUXILIAR, ID_FUNCIONARIO, NOMBRE_AUXILIAR
FROM (SELECT ID_PROCE, ID_ACTIVO, FECHA_PROCEDIMIENTO FECHA_REVISION, ESTADO_REVISION ESTADO_ACTIVO, OBSERVACIONES,
ID_AUXILIAR, TIPO_PROCEDIMIENTO FROM PROCEDIMIENTO WHERE TIPO_PROCEDIMIENTO = 'R') INNER JOIN
(SELECT ID_FUNCIONARIO, NOMBRE_FUNCIONARIO NOMBRE_AUXILIAR FROM FUNCIONARIO) ON ID_AUXILIAR = ID_FUNCIONARIO
```

Gestor Logístico de Activo (Localizador de Activos)

Se trata de una página con una serie de informes interactivos interesantes para el equipo administrativo de logística, especialmente para el personal de Servicios Generales, con el fin de coordinar traslados de activos cuando se necesite y determinar la disponibilidad de ciertos espacios en caso de eventos o actividades que se tengan que realizar en la universidad:

El primer informe es el Localizador de Activos, la cual es una tabla que enfatiza en la ubicación física que tiene cada uno de los activos, apoyándose en operaciones de JOIN para mostrar valores diferentes a las llaves foráneas.

Localizador de Activos

Q ▾		Ir		Acciones ▾								
ID Activo	Nombre Activo	ID Locacion	Nombre Locacion	Nombre Edificio	Nombre Sede	Referencia Activo	Marca Activo	Material Activo	Color Activo	Tamano Activo	Componentes Activo	
1361	Escritorio	53002	Oficina J53002	Bloque J	Sede Principal	R9RK1495J	Lexington	Metálico	Blanco	1,20 m x 0,80 m		
1362	Silla Giratoria	53002	Oficina J53002	Bloque J	Sede Principal	GR5QRVC329	Ibaby	Cuero	Negro			
1363	Computador de Escritorio	53002	Oficina J53002	Bloque J	Sede Principal	XE8IBEYEFPDF6FWZS6K5	Toshiba			21 pulgadas		
1364	Escritorio	53002	Oficina J53002	Bloque J	Sede Principal	Z28WNNA0	Lexington	Madera	Blanco	1,20 m x 0,80 m		
1293	Silla Giratoria	55001	Oficina C55001	Bloque C	Sede Principal	KSUBUZ3LT6	Ibaby	Sintético	Gris			
1294	Computador de Escritorio	11001	Oficina I11001	Bloque I	Sede Principal	P6VTAYS4ZSQJ9G1MBNP	Acer			22 pulgadas		
1295	Escritorio	11001	Oficina I11001	Bloque I	Sede Principal	TH9PV47MJ	Flexa	Metálico	Marrón	1,10 m x 0,60 m		
1296	Silla Giratoria	11001	Oficina I11001	Bloque I	Sede Principal	OTLTXI902Q	CashOffice	Malla	Negro			
1297	Computador de Escritorio	55001	Oficina C55001	Bloque C	Sede Principal	K4CWVNJWPS6RXYOQKOPJP	Asus			23 pulgadas		
1298	Escritorio	55001	Oficina C55001	Bloque C	Sede Principal	3GGXV4POK	B&B	Metálico	Marrón	1,20 m x 0,80 m		
1299	Silla Giratoria	55001	Oficina C55001	Bloque C	Sede Principal	JDS613DXWZ	CashOffice	Sintético	Gris			
1300	Computador de Escritorio	12001	Oficina L12001	Bloque Laboratorios	CIDT Pinares de Tenjo	3H59WNQPYEUJ20RED3HK	Acer			25 pulgadas		
1301	Escritorio	12001	Oficina L12001	Bloque Laboratorios	CIDT Pinares de Tenjo	DR6IFRKRC	Lexington	Madera	Blanco	1,20 m x 0,75 m		
1302	Silla Giratoria	12001	Oficina L12001	Bloque Laboratorios	CIDT Pinares de Tenjo	PKXJUCWE6PZ	SongMics	Cuero	Gris			

La consulta asociada a este objeto es la siguiente:

```
SELECT ID_ACTIVO, NOMBRE_ACTIVO, ID_LOCACION, NOMBRE_LOCACION, NOMBRE_EDIFICIO, NOMBRE_SEDE, REFERENCIA_ACTIVO,
MARCA_ACTIVO, MATERIAL_ACTIVO, COLOR_ACTIVO, TAMANO_ACTIVO, COMPONENTES_ACTIVO FROM ACTIVO NATURAL JOIN LOCACION
NATURAL JOIN EDIFICIO NATURAL JOIN SEDE
```

Un siguiente informe, presenta el número de sillas libres o faltantes según cuántas haya por edificio en relación a cuántos funcionarios trabajan en dicho edificio. Esto permite dar una idea al área de logística, a cuál edificio deberían dirigirse en caso de necesitar sillas para un evento:

Localizador de Activos

Sillas Giratorias Libres o Faltantes por Edificio

Número de sillas libres o faltantes por edificio según el número de empleados que trabajan en cierto edificio

<input type="text"/> Q ▾			<input type="button" value="Ir"/>	<input type="button" value="Acciones ▾"/>		
Libres	Id Edificio	Nombre Sede	Nombre Edificio	Sillas	Personas	
-1	6	Sede Principal	Bloque F	35	36	
0	14	CIDT Pinares de Tenjo	Bloque Laboratorios	9	9	
0	7	Sede Principal	Bloque G	18	18	
0	1	Sede Principal	Bloque A	36	36	
0	15	CIDT Pinares de Tenjo	Bloque Operarios	12	12	
-1	2	Sede Principal	Bloque B	26	27	
-3	8	Sede Principal	Bloque H	42	45	
-5	12	Dependencia Teusaquillo	Bloque Carlos Fernando Paredes Millán	54	59	
-1	11	Dependencia Teusaquillo	Bloque Principal	17	18	
-2	17	Dependencia Facatativá	Bloque Principal	16	18	
-1	5	Sede Principal	Bloque E	26	27	
-4	18	Dependencia Facatativá	Bloque Carrera 15	35	39	
75	4	Sede Principal	Bloque D	93	18	

La consulta asociada a este objeto es:

```

WITH sillas_por_edificio AS (SELECT ID_EDIFICIO, NOMBRE_SEDE, NOMBRE_EDIFICIO, COUNT(*) SILLAS FROM ACTIVO NATURAL
JOIN LOCACION NATURAL JOIN EDIFICIO NATURAL JOIN SEDE WHERE NOMBRE_ACTIVO = 'Silla Giratoria' GROUP BY ID_EDIFICIO,
NOMBRE_SEDE, NOMBRE_EDIFICIO),
personas_por_edificio AS (SELECT getEdificioByOficina(getOficinaByFuncionario(ID_FUNCIONARIO)) EDIFICIO, COUNT(*)
PERSONAS FROM FUNCIONARIO WHERE ID_AREA IS NOT NULL GROUP BY
getEdificioByOficina(getOficinaByFuncionario(ID_FUNCIONARIO)))
SELECT ID_EDIFICIO, NOMBRE_SEDE, NOMBRE_EDIFICIO, SILLAS, PERSONAS, (SILLAS-PERSONAS) LIBRES FROM sillas_por_edificio
SPE INNER JOIN personas_por_edificio PPE ON SPE.ID_EDIFICIO = PPE.EDIFICIO
  
```

También se decidió incluir la siguiente relación en la que se muestran los auxiliares de laboratorio asociados a cada laboratorio, lo cual podría ser de interés para el área de logística. Esto es, conocer cuál funcionario tiene la llave de cierto laboratorio por ejemplo:

Llaves de Laboratorios				
Se presenta el auxiliar que tiene la llave de cada laboratorio.				
Id Edificio	Id Funcionario	Nombre Funcionario	Id Locacion	Nombre Locacion
5	1019004007	Jaime Andrés Prieto Asensio	6	Investigación y Desarrollo Metrológico
5	1019004008	Angel Paul Beltrán López	7	Investigación, Calidad y Análisis
5	1019004010	Julián Joan Santamaría Varela	1	Química
5	1019004011	Nathaly Myriam Montes Campos	2	Clencias Biológicas
5	1019004012	Néstor Paul Usaquén Pérez	3	Física
5	1019004013	Steven José Sánchez Silva	9	Biocombustibles
9	1019004014	Alexis David Valencia Ramírez	1	Bioensayos y Fuentes de Alimentación No Convencionales
9	1019004015	Lorena Luz Espinoza Salas	2	Electrónica
9	1019004016	César Andrés Rubio Murillo	3	Geomática
9	1019004017	Nicol Gina Molina Álvarez	4	Hidráulica

La consulta asociada a este informe es:

```
SELECT ID_FUNCIONARIO, NOMBRE_FUNCIONARIO, ID_EDIFICIO, ID_LOCACION, NOMBRE_LOCACION FROM FUNCIONARIO F INNER JOIN LOCACION L ON F.ID_FUNCIONARIO = L.ID_AUXILIAR
```

Los siguientes dos informes que se generaron y que funcionan prácticamente igual, son dos tablas que muestran las salas de cómputo y los auditorios en los cuales hay computadores o equipos de sonido con fallas, según hayan sido revisados por auxiliares de inventarios. Esto permitirá al equipo de Logística saber cuáles espacios no están las mejores condiciones para albergar un evento dado:

Salas de Cómputo con fallas			Auditorios con fallas técnicas		
Salas de Cómputo que presentan alguna falla técnica en sus equipos			Auditorios que presentan alguna falla técnica en sus equipos		
Nombre Sede	Nombre Edificio	Nombre Locacion	Nombre Sede	Nombre Edificio	Nombre Locacion
CIDT Pinares de Tenjo	Bloque Operarios	Salón O47	Sede Principal	Bloque C	Auditorio Manuel Rafael Salas Gil
1 - 1					

Las consultas asociadas a cada uno de estos dos informes son respectivamente:

```
WITH locaciones AS (SELECT * FROM LOCACION NATURAL JOIN EDIFICIO NATURAL JOIN SEDE)
SELECT locaciones.NOMBRE_SEDE, locaciones.NOMBRE_EDIFICIO, locaciones.NOMBRE_LOCACION
FROM (SELECT * FROM ACTIVO NATURAL JOIN PROCEDIMIENTO NATURAL JOIN (SELECT ID_EDIFICIO, ID_LOCACION, NOMBRE_LOCACION,
TIPO_SALON FROM LOCACION) WHERE ESTADO_REVISION IN ('M', 'R') AND TIPO_SALON = 'COM') R
INNER JOIN locaciones ON R.ID_EDIFICIO = locaciones.ID_EDIFICIO AND R.ID_LOCACION = locaciones.ID_LOCACION
```

```

WITH locaciones AS (SELECT * FROM LOCACION NATURAL JOIN EDIFICIO NATURAL JOIN SEDE)
SELECT locaciones.NOMBRE_SEDE, locaciones.NOMBRE_EDIFICIO, locaciones.NOMBRE_LOCACION
FROM (SELECT * FROM ACTIVO NATURAL JOIN PROCEDIMIENTO NATURAL JOIN (SELECT ID_EDIFICIO, ID_LOCACION, NOMBRE_LOCACION,
TIPO_SALON FROM LOCACION) WHERE ESTADO_REVISION IN ('M', 'R') AND TIPO_SALON = 'AUD') R
INNER JOIN locaciones ON R.ID_EDIFICIO = locaciones.ID_EDIFICIO AND R.ID_LOCACION = locaciones.ID_LOCACION

```

Gestor de Puestos de Trabajo

Esta página es de interés para el personal de Gestión Humana y Desarrollo Organizacional. Presenta una serie de informes centrados en saber qué funcionarios no cuentan con un puesto de trabajo formalmente asignado. Verificar si hay funcionarios con un puesto de trabajo que presenta fallas o necesita reemplazarse. Como un registro de posibles cargos disciplinarios si hay evidencia de un uso inadecuado del activo por parte del funcionario.

Dos informes similares presentan los funcionarios sin puestos de trabajo. Uno centrado en los docentes y el otro en el personal administrativo:

Docentes sin puesto de trabajo asignado					
		Ir	Acciones		
Id Funcionario	Nombre Funcionario	Título Docente	Rol Docente	Nombre Programa	Nombre Facultad
1017005004	Alejandra Luz Domínguez Pereira	Maestría	Asociado	Administración Financiera y de Sistemas	Ciencias Económicas, Administrativas y Contables
1017005036	Lorena Mayra Redondo Exposito	Doctorado	Asociado	Licenciatura en Ciencias Naturales y Educación Ambiental	Ciencias Básicas y de la Educación
1017005063	Estela Tatiana Arias Ríos	Doctorado	Asociado	Especialización en Seguridad Industrial, Higiene y Gestión Ambiental	Ingeniería
1017005078	Juliana Yamile Rojas Macías	Doctorado	Asociado	Contaduría Pública	Ciencias Económicas, Administrativas y Contables

Administrativos sin puesto de trabajo asignado					
		Ir	Acciones		
Id Funcionario	Nombre Funcionario	Cargo Empleado	Jefe	Nombre Área	
1019004002	Yelson Snader Martínez Soto	Auxiliar	Silvia Margarita Rubio Santamaría	Compras y Almacén	
1018005215	Vicente Nicolás Suárez Reyes	Asistente	Ana Alejandra Aparicio Hurtado	Infraestructura	
1018006075	Daniel Pablo Nieto Valencia	Coordinador	Alex Julián García Bellvá	Ayudas Educativas	
1018005224	Margarita Martha Cortés Cadavid	Asistente	Patricia Angélica Santos Roldán	Ayudas Educativas	
1018006082	Isabel Margarita Guerra Jiménez	Coordinador	Román Esteban Montes Rojas	Jefatura de Derecho	
1018005240	Ana Xiomara Cabrera Cárdenas	Asistente	Adrián Andrés Quintero Meléndez	Jefatura de Ingeniería Civil	
1018006055	Ricardo Felipe Pinilla Delgado	Coordinador	Jennifer Inés Murillo Rivera	Odontología	
1018005008	Pablo Sebastián Noratto Espinoza	Asistente	Ana Ana Meléndez Mendoza	Registro y Control	
1018005177	Alejís Armando Ibáñez Flores	Asistente	Esteban Juan Montero Mora	Integración con el Egresado	
1018005033	Gonzalo Javier Montes Guerra	Asistente	Angélica Inés Montes Alvarez	Registros Calificados	
1018005045	Sandra Sonia Velásquez Camacho	Asistente	Francisco Alexis Lozano Silva	Divulgación Científica	
1018005097	Edna Yolanda Moreno Molina	Asistente Jr.	Julián Andrés Gómez Gil	Gestión de la Información Institucional	
1018006034	Ricardo Antonio Bermúdez García	Coordinador	Javier Iván Asensio Suárez	Comunicaciones	

Las consultas asociadas a cada informe respectivamente son:

```

SELECT ID_FUNCIONARIO, NOMBRE_FUNCIONARIO, TITULO_DOCENTE, ROL_DOCENTE, NOMBRE_PROGRAMA, NOMBRE_FACULTAD FROM
FUNCIONARIO NATURAL JOIN FACULTAD NATURAL JOIN PROGRAMA WHERE ID_FUNCIONARIO NOT IN (SELECT ID_FUNCIONARIO FROM
FUNCIONARIO NATURAL JOIN ACTIVO) AND TIPO_FUNCIONARIO = 'D'

SELECT R.ID_FUNCIONARIO, R.NOMBRE_FUNCIONARIO, R.CARGO_EMPLEADO, R.JEFE, NOMBRE_AREA FROM (SELECT F.ID_FUNCIONARIO,
F.NOMBRE_FUNCIONARIO, F.TIPO_FUNCIONARIO, F.CARGO_EMPLEADO, J.NOMBRE_FUNCIONARIO JEFE, F.ID_AREA
FROM FUNCIONARIO F INNER JOIN FUNCIONARIO J ON F.ID_JEFE = J.ID_FUNCIONARIO) R INNER JOIN AREA A ON R.ID_AREA =
A.ID_AREA

WHERE R.ID_FUNCIONARIO NOT IN (SELECT ID_FUNCIONARIO FROM FUNCIONARIO NATURAL JOIN ACTIVO) AND R.TIPO_FUNCIONARIO =
'E'

```

Adicionalmente, se generaron dos informes que muestran la misma información anterior pero clasificada por el título y rol del docente como por el cargo de los administrativos:

Número de docentes sin puesto asignado por título y escalafón			Número de administrativos sin puesto asignado por cargo		
Título Docente	Rol Docente	Número	Cargo Empleado	Número	
Doctorado	Asociado	3	Auxiliar	2	
Maestría	Asociado	1	Jefe de Área	2	
		1 - 2	Asistente Jr.	3	
			Coordinador	8	
			Asistente	12	
					1 - 5

Las consultas asociadas son respectivamente:

```

SELECT TITULO_DOCENTE, ROL_DOCENTE, COUNT(*) NÚMERO FROM FUNCIONARIO
WHERE ID_FUNCIONARIO NOT IN (SELECT ID_FUNCIONARIO FROM ACTIVO NATURAL JOIN FUNCIONARIO) AND TIPO_FUNCIONARIO = 'D'
GROUP BY TITULO_DOCENTE, ROL_DOCENTE

```

```

SELECT CARGO_EMPLEADO, COUNT(*) NÚMERO FROM FUNCIONARIO
WHERE ID_FUNCIONARIO NOT IN (SELECT ID_FUNCIONARIO FROM ACTIVO NATURAL JOIN FUNCIONARIO) AND TIPO_FUNCIONARIO = 'E'
GROUP BY CARGO_EMPLEADO ORDER BY NÚMERO

```

Otros dos informes similares entre sí, muestran los funcionarios cuyo puesto de trabajo presenta un activo que necesita una reparación o que necesita un cambio, es decir, comprarse uno nuevo.

Funcionarios cuyo puesto de trabajo necesita un mantenimiento									Funcionarios cuyo puesto de trabajo necesita un reemplazo								
Id Funcionario	Nombre Funcionario	Cargo Empleado	Rol Docente	Id Activo	Nombre Activo	Marca Activo	Estado Revision	Observacion									
1018006022	Jenny Nathaly Costa Campos	Coordinador		2793	Silla Giratoria	EBS	R	Después de consultar con el personal de Servicios Generales, se determinó que no vale la pena hacer una reparación.									

Las consultas asociadas son:

```

WITH c AS (SELECT ID_ACTIVO, ID_FUNCIONARIO FROM ACTIVO NATURAL JOIN PROCEDIMIENTO WHERE ESTADO_REVISION = 'M')

```

```

SELECT DISTINCT F.ID_FUNCIONARIO, NOMBRE_FUNCIONARIO FROM c INNER JOIN FUNCIONARIO F ON c.ID_FUNCIONARIO = F.ID_FUNCIONARIO

SELECT F.ID_FUNCIONARIO, F.NOMBRE_FUNCIONARIO, F.CARGO_EMPLEADO, F.ROL_DOCENTE, ID_ACTIVO, NOMBRE_ACTIVO, MARCA_ACTIVO, ESTADO_REVISION, OBSERVACIONES

FROM (SELECT * FROM ACTIVO NATURAL JOIN PROCEDIMIENTO) R INNER JOIN FUNCIONARIO F ON R.ID_FUNCIONARIO = F.ID_FUNCIONARIO

WHERE ESTADO_REVISION = 'M' OR ESTADO_REVISION = 'R' AND OBSERVACIONES IS NOT NULL

```

Finalmente, se presenta un informe en el que se listan posibles casos de descargos, es decir, procesos disciplinarios que deben cumplirse cuando se evidencia un uso inadecuado por parte del usuario de uno de los activos de su puesto de trabajo. El personal de gestión humana encargado de estos procedimientos, inicia el proceso estipulado y determina si aplica o no la sanción.

Posibles procesos de descargos								
<input type="text"/> Ir <input type="button" value="Acciones ▾"/>								
Id Funcionario	Nombre Funcionario	Cargo Empleado	Rol Docente	Id Activo	Nombre Activo	Marca Activo	Estado Revision	Observaciones
1018006022	Jenny Nathaly Costa Campos	Coordinador		2793	Silla Giratoria	EBS	R	Después de consultar con el personal de Servicios Generales, se determinó que no vale la pena hacer una reparación.
1 - 1								

La consulta asociada es la siguiente:

```

SELECT F.ID_FUNCIONARIO, F.NOMBRE_FUNCIONARIO, F.CARGO_EMPLEADO, F.ROL_DOCENTE, ID_ACTIVO, NOMBRE_ACTIVO, MARCA_ACTIVO, ESTADO_REVISION, OBSERVACIONES

FROM (SELECT * FROM ACTIVO NATURAL JOIN PROCEDIMIENTO) R INNER JOIN FUNCIONARIO F ON R.ID_FUNCIONARIO = F.ID_FUNCIONARIO

WHERE (ESTADO_REVISION = 'M' OR ESTADO_REVISION = 'R') AND OBSERVACIONES IS NOT NULL AND (TIPO_PROCEDIMIENTO IN ('R', 'P'))

```

Gestor de Mantenimientos

La décima página de la aplicación está dirigida únicamente a los auxiliares de mantenimiento. Se compone de un formulario muy similar al de los otros gestores de procedimientos y dos informes interactivos.

El campo ID_AUXILIAR es nuevamente una lista de selección con nombre MAN_AUX y con consulta SQL asociada:

```
SELECT DISTINCT (NOMBRE_FUNCIONARIO || ' - ID: ' || ID_FUNCIONARIO) display_value, ID_FUNCIONARIO return_value FROM FUNCIONARIO WHERE TIPO_AUXILIAR = 'MAN'
```

El campo ESTADO_REVISION asociado al objeto MAN_EST cuenta con la siguiente configuración:

Valor de Visualización	Valor de Retorno
Necesita comprarse uno nuevo	R

Y agregando la siguiente especificación:

Estado del Activo Revisado

Deja este espacio en blanco si la reparación fue exitosa

El botón REGISTRAR incluye la siguiente sentencia:

```
INSERT INTO PROCEDIMIENTO VALUES (9999, TO_NUMBER(:MAN_ACT), TO_NUMBER(:MAN_AUX), 'M', SYSDATE, :MAN_EST, :MAN_OBS, null);
```

El informe interactivo que va almacenando los registros insertados se construye a partir de la siguiente consulta:

```
SELECT ID_PROCE, ID_ACTIVO, FECHA_REVISION, ESTADO_ACTIVO ESTADO_POS_MANTENIMIENTO, OBSERVACIONES, ID_AUXILIAR, ID_FUNCIONARIO, NOMBRE_AUXILIAR
```

```

FROM (SELECT ID_PROCE, ID_ACTIVO, FECHA_PROCEDIMIENTO, FECHA_REVISION, ESTADO_REVISION, ESTADO_ACTIVO, OBSERVACIONES,
ID_AUXILIAR, TIPO_PROCEDIMIENTO FROM PROCEDIMIENTO WHERE TIPO_PROCEDIMIENTO = 'M') INNER JOIN
(SELECT ID_FUNCIONARIO, NOMBRE_FUNCIONARIO, NOMBRE_AUXILIAR FROM FUNCIONARIO) ON ID_AUXILIAR = ID_FUNCIONARIO

```

También se incluye el siguiente informe en el que se muestran los activos pendientes de mantenimiento:

Activos pendientes de mantenimiento										
Q.▼		Ir	Acciones▼							
Id Activo	Nombre Activo	Id Edificio	Nombre Locacion	Marca Activo	Material Activo	Color Activo	Tamano Activo	Componentes Activo		Auxiliar Que Revisó
2794	Televisor	10	Oficina J21001	Sony				Juego de Micrófonos, Soportes, Amplificador, Mezclador, Altavoces		1019004002

La consulta asociada a este objeto es:

```

WITH c AS (SELECT A.ID_ACTIVO, NOMBRE_ACTIVO, ID_EDIFICIO, ID_LOCACION, MARCA_ACTIVO, MATERIAL_ACTIVO, COLOR_ACTIVO,
TAMANO_ACTIVO, COMPONENTES_ACTIVO, ID_AUXILIAR FROM ACTIVO A INNER JOIN PROCEDIMIENTO P ON A.ID_ACTIVO = P.ID_ACTIVO
WHERE ESTADO_REVISION = 'M')

SELECT ID_ACTIVO, NOMBRE_ACTIVO, c.ID_EDIFICIO, NOMBRE_LOCACION, MARCA_ACTIVO, MATERIAL_ACTIVO, COLOR_ACTIVO,
TAMANO_ACTIVO, COMPONENTES_ACTIVO, c.ID_AUXILIAR AUXILIAR_QUE_REVISÓ FROM c INNER JOIN LOCACION ON c.ID_EDIFICIO =
LOCACION.ID_EDIFICIO AND c.ID_LOCACION = LOCACION.ID_LOCACION

```

Gestor Contable de Activos (y Formulario de Edición y Supresión de Activos)

Esta página está dirigida al personal de Contabilidad y Finanzas. Se compone de dos informes interactivos.

El primero presenta toda la información contable de los activos según las funciones implementadas con PL/SQL.

Gestor Contable de Activos

Información Contable por Activo												
Q.▼		Ir			Acciones▼							
	Id Activo	Id Compra	Nombre Activo	Marca Activo	Fecha Compra	Dias Transcurridos	Valor Activo	Depreciacion Activo	Valor Activo Hoy	Depreciacion Acumulada	Valor Depreciado	
	1361	2020015	Escritorio	Lexington	20/04/2020	94	266060	8	260579	2,06	5481	
	1362	2020015	Silla Giratoria	Ibaby	20/04/2020	94	164145	8	160764	2,06	3381	
	1363	2020018	Computador de Escritorio	Toshiba	09/02/2020	165	2730920	10	2607482	4,52	123438	
	1364	2020018	Escritorio	Lexington	09/02/2020	165	289636	7	280484	3,16	9152	
	1293	2020044	Silla Giratoria	Ibaby	10/04/2020	104	255410	8	249587	2,28	5823	
	1294	2020022	Computador de Escritorio	Acer	17/02/2020	157	2779655	10	2660130	4,3	119525	
	1295	2020022	Escritorio	Flexa	17/02/2020	157	297382	7	288431	3,01	8951	
	1296	2020022	Silla Giratoria	CashOffice	17/02/2020	157	209791	8	202574	3,44	7217	
	1297	2020002	Computador de Escritorio	Asus	18/06/2020	35	2970606	10	2942068	,96	28518	
	1298	2020002	Escritorio	B&B	18/06/2020	35	243993	7	242358	,67	1635	
	1299	2020002	Silla Giratoria	CashOffice	18/06/2020	35	322258	8	319777	,77	2481	

El informe se generó basado en un formulario para posibilitar la edición, pero se hicieron las siguientes modificaciones.

Se eliminó el botón de CREAR registro.

Se eliminaron todos los campos del formulario generado en la página (12) de edición salvo el campo DEPRECIACION_ACTIVO, el cual es un atributo que puede ser cambiado por el equipo de este departamento si así lo consideran.

También se conservó el botón SUPRIMIR, en caso de que el personal de este departamento considere que un registro no es un activo de interés para la universidad.

La consulta asociada al informe es:

```
SELECT ID_ACTIVO ID, ID_ACTIVO, ID_COMPRA, NOMBRE_ACTIVO, MARCA_ACTIVO, FECHA_COMPRA, DIAS_DESDE_COMPRA(FECHA_COMPRA)
DIAS_TRANSCURRIDOS, VALOR_ACTIVO,
DEPRECIACION_ACTIVO, VALOR_ACTIVO_HOY(VALOR_ACTIVO, DEPRECIACION_ACTIVO, FECHA_COMPRA) VALOR_ACTIVO_HOY,
DEPREC_TRANS(FECHA_COMPRA, DEPRECIACION_ACTIVO) DEPRECIACION_ACUMULADA,
(VALOR_ACTIVO - VALOR_ACTIVO_HOY(VALOR_ACTIVO, DEPRECIACION_ACTIVO, FECHA_COMPRA)) VALOR_DEPRECIADO
FROM ACTIVO NATURAL JOIN COMPRA
```

También se incluyó un segundo informe que se llamó Auditoría de Compras, en la que el personal de Contabilidad y Finanzas pueden examinar en detalle compras que podrían considerar ‘sospechosas’ o por lo menos muy grandes comparadas al promedio del valor de las compras que se hacen regularmente.

	Id Activo	Id Compra	Nombre Activo	Marca Activo	Fecha Compra	Valor Activo	Valor Compra	Referencia Activo
	924	2020001	Computador de Escritorio	HP	12/04/2020	2516213	141805710	V90310U84CHLAP7NGEUZ
	925	2020001	Computador de Escritorio	Toshiba	12/04/2020	2590149	141805710	OHX9V00B5K5GZRUKGAR0
	926	2020001	Computador de Escritorio	Lenovo	12/04/2020	1955497	141805710	QPW6KVROCSAHHBLU855
	927	2020001	Computador de Escritorio	Lenovo	12/04/2020	2178655	141805710	FKE5DHBAX5VE5ON3W4EP
	928	2020001	Computador de Escritorio	Dell	12/04/2020	2776375	141805710	E61XAMRA0V7DBN56OHTL
	929	2020001	Computador de Escritorio	HP	12/04/2020	1984413	141805710	9YVFKN81ZHBN11PMDZU
	930	2020001	Computador de Escritorio	Acer	12/04/2020	2978310	141805710	VMIW7W47PU558TNG6WWC
	931	2020001	Computador de Escritorio	Asus	12/04/2020	2131542	141805710	CEIQUN3H061GB80M6B1
	932	2020001	Computador de Escritorio	Lenovo	12/04/2020	2795214	141805710	J5MMKH1CMX390NOM2FU
	933	2020001	Computador de Escritorio	Asus	12/04/2020	1918966	141805710	BT4BG3Q00MYOLHGA1BXV
	741	2020001	Pupitre	B&B	12/04/2020	214968	141805710	26UE68UM4
	744	2020001	Pupitre	Ceccotti	12/04/2020	213796	141805710	PTZBQ2RNZ

La consulta asociada a este informe es:

```
SELECT ID_COMPRA, VALOR_COMPRA, FECHA_COMPRA, ID_ACTIVO, REFERENCIA_ACTIVO, NOMBRE_ACTIVO, MARCA_ACTIVO, VALOR_ACTIVO
FROM COMPRA NATURAL JOIN ACTIVO WHERE (VALOR_COMPRA - (SELECT AVG(VALOR_COMPRA) FROM COMPRA)) >= 25000000 ORDER BY
ID_COMPRA
```

REFERENCIAS

- [1] A. Silberchatz, H. Korth and S. Sudarshan, *Fundamentos de bases de datos*. Madrid: McGraw-Hill, 2002.
- [2] M. Qaium, "Download Text File With Unlimited Character", *Qaiumer.blogspot.com*, 2020. [Online]. Available: <http://qaiumer.blogspot.com/2018/05/download-text-file-with-unlimited.html>. [Accessed: 22- Jul-2020].