Actividad 1

Universidad Internacional de Valencia.

Maestría Oficial en Desarrollo de Aplicaciones y Servicios Web.

Bases de datos.

Edgar Gamarra

William Forero

2021.

Tabla de contenido

1. Introducción:

El Colegio Carlos Arango Vélez es una institución ubicada en la ciudad de Bogotá. Imparte la enseñanza de forma gratuita a los estudiantes en básica primaria y secundaria en horario de jornada única, el colegio tiene solo una sede.

2. Necesidades y requisitos del usuario

Cliente:

En la entrevista realizada se llega a la conclusión que se necesita una base de datos que permita almacenar la información del colegio, la entrevista se realiza con el director ya que es el que conoce toda la mecánica en el centro de estudio.

¿Para qué necesita la bbdd?

La Institución no cuenta con un sistema informático que permita optimizar sus actividades, por lo cual necesita gestionar a sus alumnos, padres, docentes, directivos, cursos, aulas, asignaturas y matrículas; llevar un seguimiento y control de accesos. Evitando de esta manera seguir manejando papelería u hojas de Excel que traen muchas veces problemas en duplicidad de la información, ya que no se tiene estandarizado el manejo de una sola herramienta para el almacenamiento, esto produce falta de organización de la información, retrasos y carga operacional al personal administrativo de la Institución. Con esta información ya almacenada se podrán entregar informes al rector referente a cuántos estudiantes se están matriculando cada año, cantidad de estudiantes por asignatura, para poder tomar decisiones en cuanto la contratación de más profesores en caso de que las aulas estén al máximo de su capacidad.

Como guardan la información

Actualmente la almacenan en hojas de Excel y hojas de papel que son archivadas en carpetas.

¿Qué operaciones realiza el Colegio?

Tarea	Descripción
Consultar, modificar y registrar estudiantes	Nombres, Apellido Paterno, Apellido Materno, Identificación (DNI), Sexo, Fecha de nacimiento, Lugar de nacimiento, Dirección, Teléfono fijo, celular, Número de matrícula, Institución de procedencia de la matrícula, Estado del alumno, Resolución Directoral, observación, lengua materna, usuario, password.
Consultar, modificar y registrar los profesores	Apellido Paterno, Apellido Materno, Nombres, Identificación (DNI), Sexo, Fecha de nacimiento, Dirección, Teléfono fijo, celular, correo institucional, nivel de estudios, salario, usuario, password.
Consultar, crear y asignar asignatura	Id de asignatura, nombre de la asignatura.
Consultar, crear y asignar nivel	Id nivel, Año, nombre del nivel.
Consultar, crear y asignar grado	Id grado, nombre del grado.
Consultar, crear y asignar sección	Id sección, nombre de la sección.
Consultar y modificar periodos	Id periodo, nombre del periodo.
Consultar, modificar y registrar notas	id, nota, fecha, id periodo.
Consultar, modificar y registrar directivo	DNI, Apellidos, nombres, cargo, usuario, password, salario, sexo, correo institucional, dirección, fecha de nacimiento, estado.
Consultar, crear y editar Matrícula	Id matrícula, estado matricula, fecha inicio matrícula, fecha finalización, turno, número de orden, Edad, observación, acudiente, Institución de procedencia
Consultar aulas	
Consultar, modificar y registrar pariente	Apellido Paterno, Apellido Materno, Nombres, DNI, vive con pariente, escolaridad del pariente, teléfono fijo del pariente, celular del pariente, ocupación del pariente, tipo de pariente, usuario, fecha de nacimiento, password.
Consultar, modificar y registrar asignatura	Id asignatura, Nombre de asignatura
Consultar, registrar, modificar área	Id área, nombre área

Requisitos funcionales finales

Requisitos	Descripción
Consultar estudiantes	Nombres, Apellido Paterno, Apellido Materno, Identificación (DNI), Sexo, Edad, Fecha de nacimiento, turno, Lugar de nacimiento, Dirección, Teléfono, Institución de procedencia, Número de matrícula, Situación de la matrícula, Estado del alumno, Resolución Directoral, observación, lengua materna, turno, nivel, grado, sección, número de orden, usuario, contraseña.
Modificar estudiantes	Nombres, Apellido Paterno, Apellido Materno, turno, Dirección, Teléfono, Situación de la matrícula, Estado del alumno, Resolución Directoral, observación, nivel, grado, sección, número de orden, usuario, contraseña.
Registrar estudiantes	Nombres, Apellido Paterno, Apellido Materno, Identificación (DNI), Sexo, Edad, Fecha de nacimiento, turno, Lugar de nacimiento, Dirección, Teléfono, Institución de procedencia, Número de matrícula, Situación de la matrícula, Estado del alumno, Resolución Directoral, observación, lengua materna, turno, nivel, grado, sección, número de orden, usuario, contraseña.
Consultar profesores	Apellido Paterno, Apellido Materno, Nombres, Identificación (DNI), Sexo, Edad, Fecha de nacimiento, Dirección, Teléfono, correo institucional, nivel de estudios, salario
Modificar profesores	Apellido Paterno, Apellido Materno, Nombres, Dirección, nivel de estudios, salario
Registrar profesores	Apellido Paterno, Apellido Materno, Nombres, Identificación (DNI), Sexo, Edad, Fecha de nacimiento, Dirección, Teléfono, correo institucional, nivel de estudios, salario
Consultar nivel	Año, descripción.
Crear nivel	Año, descripción.
Asignar nivel	Año, descripción.
Consultar grado	Año, nivel, grado.
Crear grado	Año, nivel, grado.
Asignar grado	Año, nivel, grado.
Consultar sección	Turno, nivel, grado, sección, aula.
Crear Sección	Turno, nivel, grado, sección, aula.

Asignar Sección	Turno, nivel, grado, sección, aula.
Consultar aula	Aula, disponibilidad, capacidad.
Crear aula	Aula, disponibilidad, capacidad.
Asignar aula	Aula, disponibilidad, capacidad.
Consultar periodo	Año, periodos.
Modificar periodo	Año, periodos.
Consultar notas	id estudiante, nota, curso, fecha, profesor, periodo.
Consultar matrícula	id matrícula, estado matricula, fecha inicio matrícula, fecha finalización, turno, nivel, grado, sección, número de orden, sexo, DNI del alumno, apellido paterno del alumno, apellido materno del alumno, nombres del alumno, observación, apellido paterno del acudiente, apellido materno del acudiente, nombres del acudiente, DNI del acudiente, teléfono fijo del acudiente, celular del acudiente, dirección del acudiente.
editar matrícula	estado matricula, fecha finalización, turno, nivel, grado, sección, número de orden, observación, teléfono del acudiente, dirección del acudiente.
Consultar pariente	Apellido Paterno del Padre, Apellido Materno del Padre, Nombres del Padre, DNI del padre, vive con padre, vive con padre, escolaridad del padre, teléfono del padre, celular del padre, DNI del padre, ocupación del padre, usuario del padre, contraseña del padre. Apellido paterno de la madre, Apellido Materno de la madre, Nombres de la madre, DNI de la madre, vive con madre, escolaridad de la madre, teléfono de la madre, celular de la madre, DNI de la madre, ocupación de la madre, usuario de la madre, usuario del padre. Seleccionar tipo de pariente.
Modificar pariente	vive con pariente, escolaridad del pariente, teléfono fijo del pariente, celular del pariente, ocupación del pariente, tipo de pariente, usuario del pariente, contraseña del pariente.
Registrar pariente	Apellido Paterno del pariente, Apellido Materno del pariente, Nombres del pariente, DNI del pariente, vive con pariente, escolaridad del pariente, teléfono fijo del pariente, celular del pariente, ocupación del pariente, tipo de pariente.
Consultar asignatura	Id asignatura, Nombre de asignatura
Modificar asignatura	Nombre de asignatura
Registrar asignatura	Id asignatura, Nombre de asignatura
Consultar área	Id área, nombre área

Registrar área	Id área, nombre área
Consultar directivo	Apellidos, Nombres, Identificación (DNI), Sexo, Fecha de nacimiento, Dirección, Teléfono, correo institucional, salario, cargo, estado
Modificar directivo	Apellidos, Nombres, Identificación (DNI), Sexo, Fecha de nacimiento, Dirección, Teléfono, correo institucional, salario, cargo, estado
Registrar directivo	Apellidos, Nombres, Identificación (DNI), Sexo, Fecha de nacimiento, Dirección, Teléfono, correo institucional, salario, cargo, estado

Limitar requisitos: Según la reunión establecida con el cliente y en mutuo acuerdo, se llega a la conclusión que ninguno de los requisitos indicados se queda por fuera.

3 diseño conceptual de base de datos

3.1-Para el diseño conceptual de la base de datos debemos seguir los siguientes pasos:

1. Identificar las entidades.

   Colegio, Área, Nota, Estudiante, Profesor, Directivo, Matricula, Nivel, Aula, Grado, Pariente, Asignatura, Sede, Sección, Periodo

2. Identificar las relaciones.

-Colegio Mantiene sedes, relación MANTIENE. cardinalidad máxima: Un Colegio mantiene muchas sedes y una sede es mantenida por un solo colegio. Cardinalidad mínima: Una sede es mantenida por un solo Colegio y un Colegio mantiene como mínimo una sede.

- Área fija asignaturas, relación FIJA. Cardinalidad máxima: Un área fija muchas asignaturas y una asignatura es fijada por una sola área. Cardinalidad mínima: Una asignatura es fijada por una sola área y un área fija como mínimo una asignatura.

- Área dedica profesores, relación DEDICA. Cardinalidad máxima: A un área se dedican muchos profesores y un profesor se dedica una sola área. Cardinalidad mínima: un profesor solo se dedica a un área y en un área se dedica como mínimo un profesor.

-Colegio abarca periodos, relación ABARCA. Cardinalidad máxima: Un colegio abarca muchos periodos y un periodo es abarcado por un solo Colegio. Cardinalidad mínima: Un periodo sólo es abarcado por un colegio y un Colegio abarca mínimo un periodo.

-Grado tiene sección, relación TIENE. Cardinalidad máxima un grado tiene muchas secciones y una sección tiene máximo un grado. Cardinalidad mínima: un grado tiene sólo una sección y una sección tiene mínimo un grado.

-Sección cuenta aula, relación CUENTA. Cardinalidad máxima: Una sección cuenta con un aula y una sección cuenta con un aula. Cardinalidad mínima: un aula cuenta mínimo con un aula y un aula cuentan como mínimo de una sección.

-Alumno Consigna matrícula, relación CONSIGNA. Cardinalidad máxima: Un alumno consigna una sola matrícula y una matrícula es consignada por un solo estudiante. Cardinalidad mínima: Una matrícula es consignada por mínimo 0 estudiante y un estudiante consigna mínimo 0 matrícula.

-Pariente reconoce matrícula, relación RECONOCE. Cardinalidad máxima: Un pariente reconoce muchas matrículas y una matrícula es reconocida por un pariente. Cardinalidad mínima: Una matrícula es reconocida por un pariente y un pariente reconoce mínimo 0 matrículas.

-Sede contempla nivel, relación CONTEMPLA. Cardinalidad máxima: Una sede contempla muchos niveles y un nivel es contemplado por una sede. Cardinalidad mínima: Un nivel es contemplado mínimo una sede y una sede contempla mínimo un nivel.

-Nivel incluye grado, relación INCLUYE. Cardinalidad máxima: Un nivel incluye muchos grados y un grado es incluido por un nivel. Cardinalidad mínima: un grado incluye un nivel y un nivel incluye mínimo un grado.

-Sede labora directivos, relación LABORA. Cardinalidad mínima: En una sede laboran muchos directivos y un directivo labora en una sede. Cardinalidad mínima: un directivo labora en una sede y una sede tiene mínimo un directivo.

-Profesor registra notas, relación REGISTRA. Cardinalidad máxima: un profesor registra muchas notas y una nota es registrada por un profesor. Cardinalidad mínima: Una nota es registrada como mínimo por 0 profesor y un profesor registra como mínimo 0 notas.

-Profesor Imparte asignatura, relación IMPARTE. Cardinalidad máxima: un profesor imparte, muchas asignaturas y una asignatura es impartida por muchos profesores. Cardinalidad mínima: Una asignatura es impartida por mínimo un profesor y un profesor mínimo imparte una asignatura.

-Estudiante Obtiene notas, relación OBTIENE. Cardinalidad máxima: Un estudiante obtiene muchas notas y una nota es obtenida a un solo estudiante. Cardinalidad mínima: Una nota sólo es obtenida para 0 estudiante y un estudiante obtiene como mínimo 0 notas.

-Asignatura expresa notas, relación EXPRESA. Cardinalidad máxima: una asignatura expresa muchas notas y una nota es expresada por una asignatura. Cardinalidad mínima: una nota es expresada por mínimo 0 asignaturas y una asignatura es expresada por mínimo 0 notas.

3. Identificar los atributos y asociarlos a entidades y relaciones.

Entidad 1: Colegio; Atributos: Ruc, nombre, dirección

Entidad 2: Sede; Atributos: id_sede, nombre

Entidad 3: Periodo; Atributos: id_periodo, nombre

Entidad 4: Nivel; Atributos: id_nivel, cano, nombre

Entidad 5: Directivo; Atributos: id_directivo, nombres, ape_mat, ape_pat, direccion, cargo, fecha_nac, lugar_nac, estado, salario, sexo, fecha_nac, usuario, password

Entidad 6: Grado; Atributos: id_grado, nombre

Entidad 7: Asignatura; Atributos: id_asignatura, nombre

Entidad 8: Profesor; Atributos: dni_profesor, nombres, ape_mat, ape_pat, salario, sexo, lugar_nac, direccion, nivel_estudios, celular, email, fijo, estado

Entidad 9: área; Atributos: nombre, id_area

Entidad 10: Sección; Atributos: id_seccion, nombre

Entidad 11: Matricula; Atributos: id_matricula, observacion, estado, nro_orden, turno, fecha_inicio, fecha_final, inst_proc, edad

Entidad 12: Estudiante; Atributos: dni_alumno, ape_mat, ape_pat, nombres, sexo, reso_direct, observacion, num_matri, lengua_mat, estado, fijo, lugar_nac, fecha_nac, sit_matric, celular, fijo, direccion, usuario, password

Entidad 13: Nota; Atributos: id_nota, fecha, nota, id_periodo

Entidad 14: Pariente; Atributos: dni_pariente, ocupacion, ape_pat, ape_mat, tipo_pariente, fijos, nombres, escolaridad, vive_con, fecha_nac, celular, usuario, password

Entidad 15: Aula; id_aula, nombre, capacidad, disponibilidad

4. Determinar los dominios de los atributos.

Directivo: estado (retirado, activo, deshabilitado)

Profesor: estado (retirado, activo, deshabilitado)

Matrícula: estado (activa, suspendida)

Estudiante: estado (Activo, Suspendido, retirado, Trasladado, Graduado)

Acudiente: estado (Si, No)

5. Determinar los identificadores.

Colegio: Ruc

Sede: id_sede

Periodo: id_periodo

Nivel: id_nivel

Directivo: id_directivo

Grado: id_grado

Asignatura: id_asignatura

Profesor: dni_profesor

Área: id_area

Sección: id_seccion

Matrícula: id_matricula

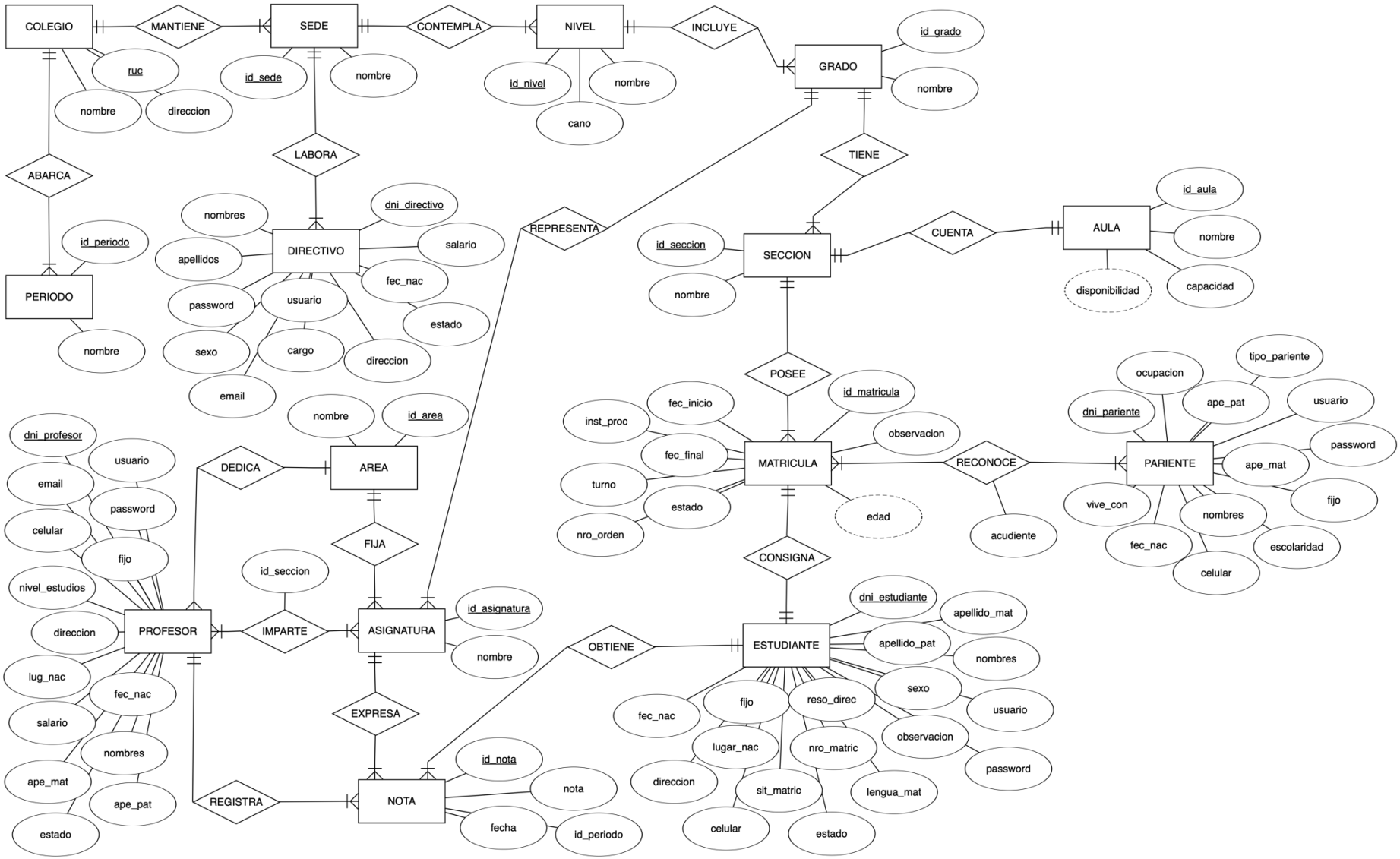
Estudiantes: dni_alumno

Nota: id_nota

Pariente: dni_pariente

Aula: id_aula

6. Dibujar el diagrama entidad-relación



3.2 Diseño lógico

3.2.1 Las entidades las transformamos a tablas

COLEGIO (ruc, nombre, dirección)

SEDE (id_sede, nombre, ruc)

PERIODO (id_periodo, nombre, ruc)

NIVEL (id_nivel, cano, nombre, id_sede)

GRADO (id_grado, nombre, id_nivel)

ASIGNATURA (id_asignatura, nombre, id_grado, id_area, dni_profesor)

PROFESOR (dni_profesor, nombre, ape_mat, ape_pat, salario, sexo, lugar_nac, direccion, nivel_estudios, celular, email, fijo, estado, id_area)

AREA (id_area, nombre)

SECCION (id_seccion, nombre, id_grado)

MATRICULA (id_matricula, observacion, estado, nro_orden, turno, fecha_inicio, fecha_final, inst_proc, edad, id_seccion, dni_estudiante)

ESTUDIANTE (dni_alumno, ape_mat, ape_pat, nombre, sexo, reso_direct, observacion, num_matri, lengua_mat, estado, fijo, lugar_nac, fecha_nac, sit_matric, celular, fijo, direccion, usuario, password)

NOTA (id_nota, fecha, nota, id_periodo)

PARIENTE (dni_pariente, ocupacion, ape_pat, ape_mat, tipo_pariente, fijo, nombre, escolaridad, vive_con, fecha_nac, celular, usuario, password)

AULA (id_aula, nombre, capacidad, disponibilidad)

DIRECTIVO (id_directivo, nombre, ape_mat, ape_pat, direccion, cargo, fecha_nac, lugar_nac, estado, salario, sexo, fecha_nac, usuario, password, id_sede)

3.2.2 Las relaciones

M a N -> Se convierten en una tabla

RECONOCE (id_matricula, dni_pariente, acudiente)

CP (id_matricula, dni_pariente)

CAj (RECONOCE.id_matricula es clave ajena a MATRICULA.id_matricula)

¿Acepta nulos?: NO

Borrado: Restringir

Restringir: MATRICULA no se puede borrar si existe fila en RECONOCE

Modificación: Restringir

Propagar: MATRICULA no se puede modificar si existe fila en RECONOCE

CAj (RECONOCE.dni_pariente es clave ajena a PARIENTE.dni_pariente)

¿Acepta nulos?: NO

Borrado: Restringir

Restringir: PARIENTE no se puede borrar si existe fila en RECONOCE

Modificación: Propagar

Propagar: dni_pariente se modifica y modifica la fila RECONOCE

IMPARTE (dni_profesor, id_asignatura, id_seccion)

CP (dni_profesor, id_asignatura, id_seccion)

CAj (IMPARTE.dni_profesor es clave ajena a PROFESOR.dni_profesor)

¿Acepta nulos?: NO

Borrado: Restringir

Restringir: PROFESOR no se puede borrar si existe fila en IMPARTE

Modificación: Restringir

Propagar: PROFESOR no se puede modificar si hay fila en IMPARTE

CAj (RECONOCE.dni_pariente es clave ajena a PARIENTE.dni_pariente)

¿Acepta nulos?: NO

Borrado: Restringir

Restringir: PARIENTE no se puede borrar si hay fila en RECONOCE

Modificación: Propagar

Propagar: dni_pariente se modifica y modifica la fila RECONOCE

3.2.3 Normalización

3.2.3.1 1FN

Se encuentran en primera forma normal

3.2.3.2 2FN

TABLA ORIGINAL:

DIRECTIVO (id_directivo, nombres, ape_mat, ape_pat, direccion, cargo, fecha_nac, lugar_nac, estado, salario, usuario, password)

PROFESOR (dni_profesor, nombres, ape_mat, ape_pat, salario, sexo, lugar_nac, direccion, nivel_estudios, celular, email, fijo, estado, id_area)

ESTUDIANTE (dni_alumno, ape_mat, ape_pat, nombres, sexo, reso_direct, observacion, num_matri, lengua_mat, estado, fijo, lugar_nac, fecha_nac, sit_matric, celular, fijo, direccion, usuario, password)

TABLA NORMALIZADA

DIRECTIVO (id_directivo, nombres, ape_mat, ape_pat, direccion, fecha_nac, estado, salario, usuario, password, id_cargo, lug_nacimiento_id_ciudad)

PROFESOR (dni_profesor, nombres, ape_mat, ape_pat, sexo direccion, nivel_estudios, celular, email, fijo, estado, id_cargo, id_area, lugar_nac_id_ciudad)

CARGO (id_cargo, nombre_cargo, salario)

CIUDADES(id_ciudad, ciudad)

3.2.3.3 3FN

Se encuentran en tercera forma normal

3.2.3.4 Desnormalización

3.2.4 Diseño Lógico Final

```
COLEGIO (ruc, nombre, dirección)
```

```
CP: {ruc}
CAj: {}
```

```
SEDE (id_sede, nombre, ruc)
```

```
CP: {id_sede}
CAj: {COLEGIO.ruc es clave ajena a SEDE.ruc}
¿Acepta nulos?: NO
Borrado: Restringir
Restringir: COLEGIO no se puede borrar si existe fila en SEDE
Modificación: Propagar
         Propagar: ruc se modifica y modifica la fila SEDE
```

PERIODO (id_periodo, nombre, ruc)

CP: {id_periodo}
CAj: {COLEGIO.ruc es clave ajena a PERIODO.ruc}
¿Acepta nulos?: NO
Borrado: Restringir
Restringir: COLEGIO no se puede borrar si existe fila en PERIODO
Modificación: Propagar
Propagar: ruc se modifica y modifica la fila PERIODO

NIVEL (id_nivel, cano, nombre, id_sede)

CP: {id_nivel}
CAj: {NIVEL.id_sede es clave ajena a SEDE.id_sede}
¿Acepta nulos?: NO
Borrado: Restringir
Restringir: SEDE no se puede borrar si existe fila en NIVEL
Modificación: Propagar
Propagar: id_sede se modifica y modifica la fila NIVEL

GRADO (id_grado, nombre, id_nivel)

CP: {id_grado}
CAj: {NIVEL.id_nivel es clave ajena a GRADO.id_nivel}
¿Acepta nulos?: NO
Borrado: Restringir
Restringir: NIVEL no se puede borrar si existe fila en GRADO
Modificación: Propagar
Propagar: id_nivel se modifica y modifica la fila GRADO

ASIGNATURA (id_asignatura, nombre, id_grado, id_area, dni_profesor)

CP: {id_asignatura}
CAj: {ASIGNATURA.id_grado es clave ajena a GRADO.id_grado}
¿Acepta nulos?: NO
Borrado: Restringir
Restringir: GRADO no se puede borrar si existe fila en ASIGNATURA
Modificación: Propagar
Propagar: id_grado se modifica y modifica la fila ASIGNATURA
CAj: {ASIGNATURA.id_area es clave ajena a AREA.id_area}
¿Acepta nulos?: NO
Borrado: Restringir
Restringir: AREA no se puede borrar si existe fila en ASIGNATURA
Modificación: Propagar
Propagar: id_area se modifica y modifica la fila ASIGNATURA
CAj: {ASIGNATURA.dni_profesor es clave ajena a PROFESOR.dni_profesor}
¿Acepta nulos?: NO
Borrado: Restringir
Restringir: PROFESOR no se puede borrar si existe fila en ASIGNATURA
Modificación: Propagar
Propagar: dni_profesor se modifica y modifica la fila ASIGNATURA

PROFESOR (dni_profesor, nombres, ape_mat, ape_pat, id_cargo, sexo, id_ciudad, direccion, nivel_estudios, celular, email, fijo, estado, id_area)

CP: {dni_profesor}
CAj: {PROFESOR.id_area es clave ajena a AREA.id_area}
¿Acepta nulos?: NO
Borrado: Restringir
Restringir: AREA no se puede borrar si existe fila en PROFESOR
Modificación: Propagar
Propagar: id_area se modifica y modifica la fila PROFESOR
CAj: {PROFESOR.id_cargo es clave ajena a CARGO.id_cargo}
¿Acepta nulos?: NO
Borrado: Restringir
Restringir: CARGO no se puede borrar si existe fila en PROFESOR
Modificación: Propagar
Propagar: id_cargo se modifica y modifica la fila PROFESOR
CAj: {PROFESOR.id_ciudad es clave ajena a CIUDAD_NACIMIENTO.id_ciudad}
¿Acepta nulos?: NO
Borrado: Restringir
Restringir: CIUDAD no se puede borrar si existe fila en PROFESOR
Modificación: Propagar

```
          Propagar: id_ciudad se modifica y modifica la fila PROFESOR
```

```
Dominio de estado: [retirado, activo, deshabilitado]
```

```
AREA (id_area, nombre)
```

```
CP: {id_area}
CAj: {}
```

```
SECCION (id_seccion, nombre, id_grado)
```

```
CP: {id_seccion}
CAj: {SECCION.id_grado es clave ajena a GRADO.id_grado}
¿Acepta nulos?: NO
Borrado: Restringir
Restringir: GRADO no se puede borrar si existe fila en SECCION
Modificación: Propagar
          Propagar: id_grado se modifica y modifica la fila SECCION
```

```
MATRICULA (id_matricula, observacion, estado, nro_orden, turno, fecha_inicio, fecha_final,
inst_proc, edad, id_seccion, dni_estudiante)
```

```
CP: {id_matricula}
CAj: {MATRICULA.id_seccion es clave ajena a SECCION.id_seccion}
¿Acepta nulos?: NO
Borrado: Restringir
Restringir: SECCION no se puede borrar si existe fila en MATRICULA
Modificación: Propagar
          Propagar: id_seccion se modifica y modifica la fila MATRICULA
CAj: {MATRICULA.dni_estudiante es clave ajena a ESTUDIANTE.id_estudiante}
¿Acepta nulos?: NO
Borrado: Restringir
Restringir: ESTUDIANTE no se puede borrar si existe fila en MATRICULA
Modificación: Propagar
          Propagar: dni_estudiante se modifica y modifica la fila MATRICULA
Dominio de estado: [activa, suspendida]
```

```
ESTUDIANTE (dni_alumno, ape_mat, ape_pat, nombres, sexo, reso_direct, observacion, num_matri,
lengua_mat, estado, fijo, id_ciudad, fecha_nac, sit_matric, celular, direccion, usuario,
password)
```

```
CP: {dni_alumno}
CAj: {ESTUDIANTE.id_ciudad es clave ajena a CIUDAD_NACIMIENTO.id_ciudad}
¿Acepta nulos?: NO
Borrado: Restringir
Restringir: CIUDAD no se puede borrar si existe fila en ESTUDIANTE
Modificación: Propagar
          Propagar: id_ciudad se modifica y modifica la fila ESTUDIANTE
```

```
Dominio de estado: [activo, suspendido, retirado, trasladado, graduado]
```

```
NOTA (id_nota, fecha, nota, id_periodo, dni_profesor, dni_estudiante, id_asignatura)
```

```
CP: {id_nota}
CAj: {NOTA.dni_profesor es clave ajena a PROFESOR.id_profesor}
¿Acepta nulos?: NO
Borrado: Restringir
Restringir: PROFESOR no se puede borrar si existe fila en NOTA
Modificación: Propagar
          Propagar: dni_profesor se modifica y modifica la fila NOTA
CAj: {NOTA.dni_estudiante es clave ajena a ESTUDIANTE.dni_estudiante}
¿Acepta nulos?: NO
Borrado: Restringir
Restringir: ESTUDIANTE no se puede borrar si existe fila en NOTA
Modificación: Propagar
          Propagar: dni_estudiante se modifica y modifica la fila NOTA
CAj: {NOTA.id_asignatura es clave ajena a ASIGNATURA.id_asignatura}
¿Acepta nulos?: NO
Borrado: Restringir
Restringir: ASIGNATURA no se puede borrar si existe fila en NOTA
Modificación: Propagar
```

```
          Propagar: id_asignatura se modifica y modifica la fila NOTA
```

```
PARIENTE (dni_pariente, ocupacion, ape_pat, ape_mat, tipo_pariente, fijo, nombres, escolaridad,
vive_con, fecha_nac, celular, usuario, password)
```

```
CP: {dni_pariente}
CAj: {}
```

```
AULA (id_aula, nombre, capacidad, disponibilidad)
```

```
CP: {id_aula}
CAj: {}
```

```
DIRECTIVO (id_directivo, nombres, ape_mat, ape_pat, direccion, fecha_nac, lugar_nac, estado,
salario, sexo, fecha_nac, usuario, password, id_sede, id_cargo)
```

```
CP: {id_directivo}
CAj: {DIRECTIVO.id_sede es clave ajena a SEDE.id_sede}
¿Acepta nulos?: NO
Borrado: Restringir
Restringir: SEDE no se puede borrar si existe fila en DIRECTIVO
Modificación: Propagar
          Propagar: id_sede se modifica y modifica la fila DIRECTIVO
CAj: {DIRECTIVO.id_cargo es clave ajena a CARGO.id_cargo}
¿Acepta nulos?: NO
Borrado: Restringir
Restringir: CARGO no se puede borrar si existe fila en DIRECTIVO
Modificación: Propagar
          Propagar: id_cargo se modifica y modifica la fila DIRECTIVO
Dominio de estado: [retirado, activo, deshabilitado]
```

```
CARGO (id_cargo, nombre_cargo, salario)
```

```
CP: {id_cargo}
CAJ: {}
```

```
RECONOCE (id_matricula, dni_pariente, acudiente)
```

```
CP: (id_matricula, dni_pariente)
CAj (RECONOCE.id_matricula es clave ajena a MATRICULA.id_matricula)
¿Acepta nulos?: NO
Borrado: Restringir
Restringir: MATRICULA no se puede borrar si existe fila en RECONOCE
Modificación: Restringir
Propagar: MATRICULA no se puede modificar si existe fila en RECONOCE

CAj (RECONOCE.dni_pariente es clave ajena a PARIENTE.dni_pariente)
¿Acepta nulos?: NO
Borrado: Restringir
Restringir: PARIENTE no se puede borrar si existe fila en RECONOCE
Modificación: Propagar
          Propagar: dni_pariente se modifica y modifica la fila RECONOCE
Dominio de acudiente: [Si, No]
```

```
IMPARTE (dni_profesor, id_asignatura, id_seccion)
```

```
CP (dni_profesor, id_asignatura, id_seccion)
CAj (IMPARTE.dni_profesor es clave ajena a PROFESOR.dni_profesor)
¿Acepta nulos?: NO
Borrado: Restringir
Restringir: PROFESOR no se puede borrar si existe fila en IMPARTE
Modificación: Restringir
Propagar: PROFESOR no se puede modificar si hay fila en IMPARTE
CAj (RECONOCE.dni_pariente es clave ajena a PARIENTE.dni_pariente)
¿Acepta nulos?: NO
Borrado: Restringir
Restringir: PARIENTE no se puede borrar si hay fila en RECONOCE
Modificación: Propagar
          Propagar: dni_pariente se modifica y modifica la fila RECONOCE
```

3.3 Diseño físico

```
CREATE TABLE COLEGIO (

ruc character varying(9) NOT NULL,

nombre character varying(50) NOT NULL,

direccion character varying(50) NOT NULL,

CONSTRAINT cp_colegio_carlosarango PRIMARY KEY (ruc)

);

CREATE TABLE CIUDAD_NACIMIENTO(
id_ciudad SERIAL NOT NULL,
nombre character varying(50) NOT NULL,
CONSTRAINT cp_id_ciudad PRIMARY KEY(id_ciudad)
);

CREATE TABLE CARGO(
id_cargo serial NOT NULL,
nombre character varying(20) NOT NULL,
salario integer NOT NULL,
CONSTRAINT cp_id_cargo PRIMARY KEY(id_cargo)
);

CREATE TABLE SEDE(

id_sede serial NOT NULL,

nombre character varying(50) NOT NULL,
```

```
ruc character varying(9) NOT NULL,

CONSTRAINT cp_sede PRIMARY KEY (id_sede),

CONSTRAINT caj_colegio_sede FOREIGN KEY (ruc)

REFERENCES COLEGIO(ruc)

ON DELETE RESTRICT

ON UPDATE CASCADE

);

CREATE TABLE PERIODO (

id_periodo serial NOT NULL,

nombre character varying(50) NOT NULL,

ruc character varying(9) NOT NULL,

CONSTRAINT cp_periodo PRIMARY KEY (id_periodo),

CONSTRAINT caj_colegio_periodo FOREIGN KEY (ruc)

REFERENCES COLEGIO(ruc)

ON DELETE RESTRICT

ON UPDATE CASCADE
```

```
);

CREATE TABLE NIVEL(

id_nivel serial NOT NULL,

cano character varying(50) NOT NULL,

nombre character varying(50) NOT NULL,

id_sede integer NOT NULL,

CONSTRAINT cp_nivel PRIMARY KEY (id_nivel),

CONSTRAINT caj_sede FOREIGN KEY (id_sede)

REFERENCES SEDE(id_sede)

ON DELETE RESTRICT

ON UPDATE CASCADE

);

CREATE TABLE GRADO(

id_grado serial NOT NULL,

nombre character varying(50) NOT NULL,
```

```
id_nivel integer NOT NULL,

CONSTRAINT cp_grado PRIMARY KEY (id_grado),

CONSTRAINT caj_nivel FOREIGN KEY (id_nivel)

REFERENCES NIVEL(id_nivel)

ON DELETE RESTRICT

ON UPDATE CASCADE

);



CREATE TABLE AREA(

id_area serial NOT NULL,

nombre character varying(50) NOT NULL,

CONSTRAINT cp_area PRIMARY KEY (id_area)

);

CREATE DOMAIN estado_profesor AS character varying(15)

CHECK (VALUE IN ('retirado', 'activo', 'deshabilitado'));



CREATE TABLE PROFESOR(
```

dni_profesor integer NOT NULL,

nombre character varying(50) NOT NULL,

ape_mat character varying(50) NOT NULL,

ape_pat character varying(50) NOT NULL,

id_cargo integer NOT NULL,

sexo character varying(10) NOT NULL,

id_ciudad integer NOT NULL,

direccion character varying(50) NOT NULL,

nivel_estudios character varying(15) NOT NULL,

celular character varying(15) NOT NULL,

email character varying(30) NOT NULL,

fijo character varying(10) NOT NULL,

estado estado_profesor NOT NULL,

id_area integer NOT NULL,

CONSTRAINT cp_profesor PRIMARY KEY (dni_profesor),

```
CONSTRAINT caj_area FOREIGN KEY (id_area)

REFERENCES AREA(id_area)

ON DELETE RESTRICT

ON UPDATE CASCADE,

CONSTRAINT caj_salario_prof FOREIGN KEY (id_cargo)

REFERENCES CARGO(id_cargo)

ON DELETE RESTRICT

ON UPDATE CASCADE,

CONSTRAINT caj_lugar_prof FOREIGN KEY (id_ciudad)

REFERENCES CIUDAD_NACIMIENTO(id_ciudad)

ON DELETE RESTRICT

ON UPDATE CASCADE

);

CREATE TABLE ASIGNATURA(

id_asignatura serial NOT NULL,
```

```
nombre character varying(50) NOT NULL,

id_grado integer NOT NULL,

id_area integer NOT NULL,

dni_profesor integer NOT NULL,

CONSTRAINT cp_asignatura PRIMARY KEY (id_asignatura),

CONSTRAINT caj_grado FOREIGN KEY (id_grado)

REFERENCES GRADO(id_grado)

ON DELETE RESTRICT

ON UPDATE CASCADE,

CONSTRAINT caj_area FOREIGN KEY (id_area)

REFERENCES AREA(id_area)

ON DELETE RESTRICT

ON UPDATE CASCADE,

CONSTRAINT caj_profesor FOREIGN KEY (dni_profesor)

REFERENCES PROFESOR(dni_profesor)

ON DELETE RESTRICT
```

```
ON UPDATE CASCADE

);

CREATE TABLE SECCION(

id_seccion serial NOT NULL,

nombre character varying(50) NOT NULL,

id_grado integer NOT NULL,

CONSTRAINT cp_seccion PRIMARY KEY (id_seccion),

CONSTRAINT caj_grado FOREIGN KEY (id_grado)

REFERENCES GRADO(id_grado)

ON DELETE RESTRICT

ON UPDATE CASCADE




);

CREATE DOMAIN estado_alumno AS character varying(15)

CHECK (VALUE IN ('activo', 'suspendido', 'retirado', 'trasladado', 'graduado'));
```

```
CREATE TABLE ESTUDIANTE(

dni_alumno integer NOT NULL,

ape_mat character varying(50) NOT NULL,

ape_pat character varying(50) NOT NULL,

nombres character varying(50) NOT NULL,

sexo character varying(10) NOT NULL,

reso_direct character varying(20) NOT NULL,

observacion character varying(100) NOT NULL,

num_matri character varying(20) NOT NULL,

lengua_mat character varying(30) NOT NULL,

estado estado_alumno NOT NULL,

fijo character varying(10) NOT NULL,

id_ciudad integer NOT NULL,

fecha_nac Date NOT NULL,

sit_matric character varying(20) NOT NULL,
```

```
celular character varying(20) NOT NULL,

direccion character varying(50) NOT NULL,

usuario character varying(10) NOT NULL,

clave character varying(12) NOT NULL,

CONSTRAINT cp_estudiante PRIMARY KEY (dni_alumno),
CONSTRAINT cp_lugar_estudiante FOREIGN KEY (id_ciudad)

REFERENCES CIUDAD_NACIMIENTO(id_ciudad)

ON DELETE RESTRICT

ON UPDATE CASCADE




);


CREATE DOMAIN estado_matricula AS character varying(15)

CHECK (VALUE IN ('activa', 'suspendida'));


CREATE TABLE MATRICULA(

id_matricula serial NOT NULL,
```

```
observacion character varying(100) NOT NULL,

estado estado_matricula NOT NULL,

nro_orden character varying(20) NOT NULL,

turno character varying(10) NOT NULL,

fecha_inicio date NOT NULL,

fecha_final date NOT NULL,

inst_proc character varying(20) NOT NULL,

edad integer NOT NULL,

id_seccion integer NOT NULL,

dni_alumno integer NOT NULL,

CONSTRAINT cp_matricula PRIMARY KEY (id_matricula),

CONSTRAINT caj_seccion FOREIGN KEY (id_seccion)

REFERENCES SECCION(id_seccion)

ON DELETE RESTRICT

ON UPDATE CASCADE,
```

```
CONSTRAINT caj_estudiante FOREIGN KEY (dni_alumno)

REFERENCES ESTUDIANTE(dni_alumno)

ON DELETE RESTRICT

ON UPDATE CASCADE




);



CREATE TABLE NOTA(

id_nota serial NOT NULL,

fecha Date NOT NULL,

nota integer NOT NULL,

id_periodo integer NOT NULL,

dni_profesor integer NOT NULL,

dni_alumno integer NOT NULL,

id_asignatura integer NOT NULL,
```

```
CONSTRAINT cp_nota PRIMARY KEY(id_nota),

CONSTRAINT caj_periodo FOREIGN KEY(id_periodo)

REFERENCES PERIODO(id_periodo)

ON DELETE RESTRICT

ON UPDATE CASCADE,

CONSTRAINT caj_dni_profesor FOREIGN KEY(dni_profesor)

REFERENCES PROFESOR(dni_profesor)

ON DELETE RESTRICT

ON UPDATE CASCADE,

CONSTRAINT caj_dni_alumno FOREIGN KEY(dni_alumno)

REFERENCES ESTUDIANTE(dni_alumno)

ON DELETE RESTRICT

ON UPDATE CASCADE,

CONSTRAINT caj_id_asignatura FOREIGN KEY(id_asignatura)

REFERENCES ASIGNATURA(id_asignatura)

ON DELETE RESTRICT
```

```
ON UPDATE CASCADE



);



CREATE TABLE PARIENTE(

dni_pariente integer NOT NULL,

ocupacion character varying(50) NOT NULL,

ape_pat character varying(50) NOT NULL,

ape_mat character varying(50) NOT NULL,

nombres character varying(50) NOT NULL,

tipo_pariente character varying(15) NOT NULL,

escolaridad character varying(15) NOT NULL,

vive_con character varying(50) NOT NULL,

fecha_nac Date NOT NULL,

celular character varying(10) NOT NULL,

fijo character varying(10) NOT NULL,
```

```
usuario character varying(10) NOT NULL,

clave character varying(10) NOT NULL,

CONSTRAINT cp_dni_pariente PRIMARY KEY(dni_pariente)



);



CREATE TABLE DIRECTIVO(
id_directivo SERIAL NOT NULL,
Identificacion character varying(12) NOT NULL,
nombres character varying(50) NOT NULL,
ape_mat character varying(50) NOT NULL,
ape_pat character varying(50) NOT NULL,
direccion character varying(50) NOT NULL,
fecha_nac Date NOT NULL,
id_ciudad integer NOT NULL,
estado estado_profesor NOT NULL,
sexo character varying(10) NOT NULL,
usuario character varying(10) NOT NULL,
clave character varying(12) NOT NULL,
id_sede integer NOT NULL,
id_cargo integer NOT NULL,
CONSTRAINT cp_id_directivo PRIMARY KEY(id_directivo),
CONSTRAINT caj_id_ciudad FOREIGN KEY(id_ciudad)
REFERENCES CIUDAD_NACIMIENTO(id_ciudad)
ON DELETE RESTRICT
```

```
ON UPDATE CASCADE,
CONSTRAINT caj_id_sede FOREIGN KEY(id_sede)
REFERENCES SEDE(id_sede)
ON DELETE RESTRICT
ON UPDATE CASCADE,
CONSTRAINT caj_id_cargo FOREIGN KEY(id_cargo)
REFERENCES CARGO(id_cargo)
ON DELETE RESTRICT
ON UPDATE CASCADE

);

CREATE TABLE AULA(
id_aula SERIAL NOT NULL,
nombre character varying(50) NOT NULL,
capacidad integer NOT NULL,
CONSTRAINT cp_id_aula PRIMARY KEY(id_aula)

);


CREATE DOMAIN estado_acudiente AS character varying(15)

CHECK (VALUE IN ('Si', 'No'));


CREATE TABLE RECONOCE(

id_matricula integer NOT NULL,

dni_pariente integer NOT NULL,
```

```
acudiente estado_acudiente NOT NULL,

CONSTRAINT cp_id_matricula PRIMARY KEY(id_matricula, dni_pariente),

CONSTRAINT caj_id_matricula FOREIGN KEY(id_matricula)

REFERENCES MATRICULA(id_matricula)
ON DELETE RESTRICT
ON UPDATE CASCADE,

CONSTRAINT caj_dni_pariente FOREIGN KEY(dni_pariente)
REFERENCES PARIENTE(dni_pariente)
ON DELETE RESTRICT
ON UPDATE CASCADE

);


CREATE TABLE IMPARTE (
dni_profesor integer NOT NULL,
id_asignatura integer NOT NULL,
id_seccion integer NOT NULL,
CONSTRAINT cp_id_combinado PRIMARY KEY(dni_profesor, id_asignatura, id_seccion),
CONSTRAINT caj_imparte FOREIGN KEY(dni_profesor)
REFERENCES PROFESOR(dni_profesor)
ON DELETE RESTRICT
ON UPDATE CASCADE,
CONSTRAINT caj_imparte_asignatura FOREIGN KEY(id_asignatura)
REFERENCES ASIGNATURA(id_asignatura)
ON DELETE RESTRICT
```

ON UPDATE CASCADE,

CONSTRAINT caj_imparte_seccion FOREIGN KEY(id_seccion)

REFERENCES SECCION(id_seccion)

ON DELETE RESTRICT

ON UPDATE CASCADE

);

INSERTS

ARCHIVO INSERTS.SQL

5.1

JOINS

ARCHIVO APLICACION_BBDD.PHP

1- Obtener las notas de los alumnos

2-Con el inner join se pueden unir dos tablas para traer las notas de los estudiantes

3- select * from estudiante INNER JOIN nota ON estudiante.dni_alumno=nota.dni_alumno

1-Obtener el cargo y salario de los profesores

2-Con left join se trae la información dando prioridad a la tabla profesores

3- select * from profesor LEFT JOIN cargo ON profesor.id_cargo=cargo.id_cargo

5.2

COUNT

1-Obtener la media de notas de los estudiantes

2-con AVG podemos obtener la media de las notas

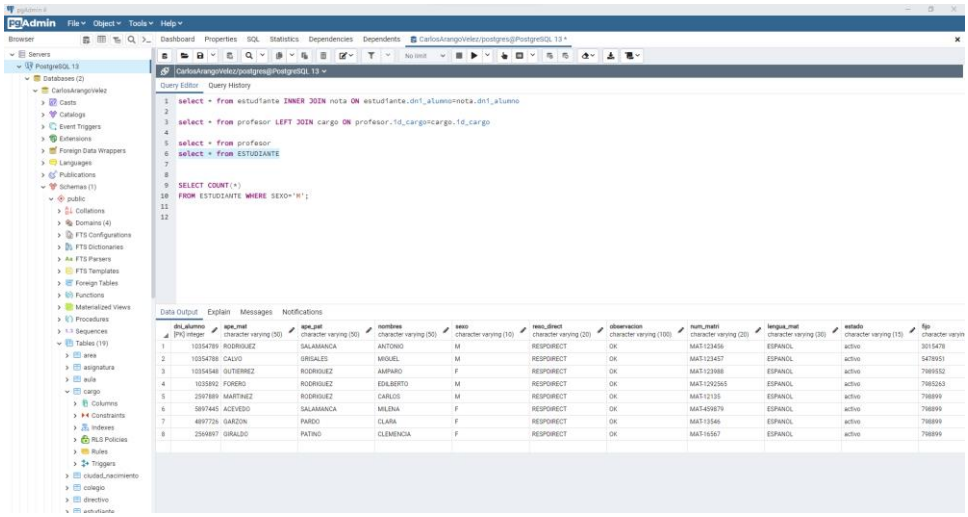
3 SELECT AVG(nota) FROM nota;

1-Obtener la cantidad de mujeres estudiantes en el colegio

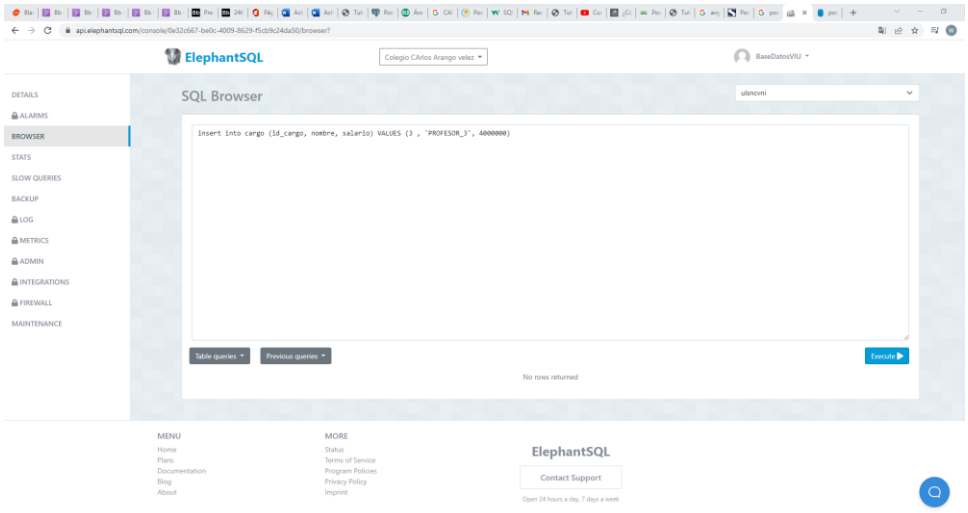
2-con COUNT podemos obtener la suma de estudiantes de acuerdo a la condición dada con la sentencia where

3- SELECT COUNT(*) FROM ESTUDIANTE WHERE SEXO='M';

POSTGREST LOCAL



POSTGREST nube



Conclusiones

La actividad ayuda fortalecer los conocimientos en consultas SQL y además se tiene una idea más clara del modelamiento de bases de datos.

Las sentencias JOIN nos permiten de alguna manera mostrar los resultados de consultas más complejas en donde se requiera saber información que con un SELECT sencillo no se puede validar.