

Nuevo Campo

Creación del campo en la BD

Nos dirigimos a MySQL y creamos el campo en la base de datos:

```
USE app_radfpd;
ALTER TABLE sgi_familias
ADD COLUMN cod_familia VARCHAR(50) NOT NULL DEFAULT "";
```

```
MariaDB [(none)]> use app_radfpd;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [app_radfpd]> ALTER TABLE sgi_familias
-> ADD COLUMN cod_familia VARCHAR(50) NOT NULL DEFAULT "";
Query OK, 0 rows affected (0.010 sec)
Records: 0 Duplicates: 0 Warnings: 0

MariaDB [app_radfpd]> SHOW COLUMNS FROM sgi_familias;
+-----+-----+-----+-----+-----+-----+
| Field      | Type      | Null | Key | Default | Extra      |
+-----+-----+-----+-----+-----+-----+
| id_familia | int(11)   | NO   | PRI | NULL    | auto_increment |
| familia    | varchar(50) | NO   |     | NULL    |              |
| observaciones | text      | YES  |     | NULL    |              |
| cod_familia | varchar(50) | NO   |     |         |              |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.001 sec)
```

Como podemos observar, se ha creado correctamente.

Modificación código

Comenzamos editando el archivo familias.component.html que se encuentra en /src/app/familias/, añadiendo un contenedor de códigos.

```
<div class="example-container mat-elevation-z8">
  <div class="example-table-container">
    <table mat-table [dataSource]="dataSource" class="mat-elevation-z8" matSort>

      <ng-container matColumnDef="id_familia">
        <th mat-header-cell *matHeaderCellDef>
          <input matInput [formControl]="idFamiliaFilter" placeholder="Id">
          <span mat-sort-header></span>
        </th>
        <td mat-cell *matCellDef="let familias"> {{familias.id_familia}} </td>
      </ng-container>

      <ng-container matColumnDef="familia">
        <th mat-header-cell *matHeaderCellDef>
          <input matInput [formControl]="familiaFilter" placeholder="Familia">
          <span mat-sort-header></span>
        </th>
        <td mat-cell *matCellDef="let familias"> {{familias.familia}} </td>
      </ng-container>
    </table>
  </div>
</div>
```

```

<ng-container matColumnDef="cod_familia">
  <th mat-header-cell *matHeaderCellDef>
    <input matInput [formControl]="codFamiliaFilter" placeholder="Código Familia">
    <span mat-sort-header></span>
  </th>
  <td mat-cell *matCellDef="let familias"> {{familias.cod_familia}} </td>
</ng-container>
...

```

Y también editamos el `familias.component.ts`, añadiendo el campo de códigos en las secciones donde vemos que están presentes los campos `id_familia` y `familia`:

```

import { Component, OnInit, ViewChild } from '@angular/core';
import { MatPaginator } from '@angular/material/paginator';
import { MatTableDataSource } from '@angular/material/table';
import { MatSort } from '@angular/material/sort';
import { MatDialog } from '@angular/material/dialog';
import { Overlay } from '@angular/cdk/overlay';
import { FormControl } from '@angular/forms';
import { Permisos } from '../shared/interfaces/api-response';
import { combineLatest } from 'rxjs';
import { startWith } from 'rxjs/operators';

import { Familia } from '../shared/interfaces/familia';
import { FamiliasService } from '../services/familias.service';

import { AddFamiliaComponent } from './add-familia/add-familia.component';
import { EditFamiliaComponent } from './edit-familia/edit-familia.component';
import { DeleteFamiliaComponent } from './delete-familia/delete-familia.component';

@Component({
  selector: 'app-familias',
  templateUrl: './familias.component.html',
  styleUrls: ['./familias.component.scss']
})
export class FamiliasComponent implements OnInit {

  @ViewChild(MatPaginator, { static: true }) paginator: MatPaginator;
  @ViewChild(MatSort, { static: true }) sort: MatSort;

  dataSource: MatTableDataSource<Familia> = new MatTableDataSource();

  idFamiliaFilter = new FormControl();
  familiaFilter = new FormControl();
  codFamiliaFilter = new FormControl();

  permisos: Permisos;

  displayedColumns: string[];
  private filterValues = { id_familia: '', familia: '', cod_familia: '' };

  constructor(
    public dialog: MatDialog,
    private familiasService: FamiliasService,
    private overlay: Overlay
  ) { }

```

```

ngOnInit(): void {
    this.getFamilias();
}

async getFamilias() {
    const RESPONSE = await this.familiasService.getAllFamilias().toPromise();
    this.permises = RESPONSE.permises;

    if (RESPONSE.ok) {
        this.familiasService.familia = RESPONSE.data as Familia[];
        this.displayedColumns = ['id_familia', 'familia', 'cod_familia', 'actions'];
        this.dataSource.data = this.familiasService.familia;
        this.dataSource.sort = this.sort;
        this.dataSource.paginator = this.paginator;
        this.dataSource.filterPredicate = this.createFilter();
        this.onChanges();
    }
}

async addFamilia() {
    const dialogRef = this.dialog.open(AddFamiliaComponent, { scrollStrategy:
this.overlay.scrollStrategies.noop() });
    const RESULT = await dialogRef.afterClosed().toPromise();
    if (RESULT) {
        if (RESULT.ok) {
            //this.familiasService.familia.push(RESULT.data);
            //this.dataSource.data = this.familiasService.familia;
            this.ngOnInit();
        }
    }
}

async editFamilia(familia: Familia) {
    const dialogRef = this.dialog.open(EditFamiliaComponent, { data: familia, scrollStrategy:
this.overlay.scrollStrategies.noop() });
    const RESULT = await dialogRef.afterClosed().toPromise();
    if (RESULT) {
        if (RESULT.ok) {
            //this.familiasService.editFamilia(RESULT.data);
            //this.dataSource.data = this.familiasService.familia;
            this.ngOnInit();
        }
    }
}

async deleteFamilia(familia: Familia) {
    const dialogRef = this.dialog.open>DeleteFamiliaComponent, { data: familia, scrollStrategy:
this.overlay.scrollStrategies.noop() });
    const RESULT = await dialogRef.afterClosed().toPromise();
    if (RESULT) {
        if (RESULT.ok) {
            //this.familiasService.deleteFamilia(RESULT.data);
            //this.dataSource.data = this.familiasService.familia;
            this.ngOnInit();
        }
    }
}

createFilter(): (familia: Familia, filter: string) => boolean {
    const filterFunction = (familia: Familia, filter: string): boolean => {
        const searchTerms = JSON.parse(filter);

```

```

        return familia.id_familia.toString().indexOf(searchTerms.id_familia) !== -1
        && familia.familia.toLowerCase().indexOf(searchTerms.familia.toLowerCase()) !== -1
        && familia.cod_familia.toString().indexOf(searchTerms.cod_familia.toLowerCase()) !== -1;
    };

    return filterFunction;
}

onChanges() {
    this.idFamiliaFilter.valueChanges
    .subscribe(value => {
        this.filterValues.id_familia = value;
        this.dataSource.filter = JSON.stringify(this.filterValues);
    });

    this.familiaFilter.valueChanges
    .subscribe(value => {
        this.filterValues.familia = value;
        this.dataSource.filter = JSON.stringify(this.filterValues);
    });
    this.codFamiliaFilter.valueChanges
    .subscribe(value => {
        this.filterValues.cod_familia = value;
        this.dataSource.filter = JSON.stringify(this.filterValues);
    });
}
}

```

Y tambien en /src/app/shared/interfaces/familia.ts debemos añadir a la interfaz el apartado cod_familia:

```

export interface Familia {
    id_familia: string;
    familia: string;
    cod_familia: string;
    observaciones?: any;
}

```

Ahora para cambiar el EDIT, vamos a /src/app/familias/edit-familia y editamos el html y ts de la siguiente manera:

HTML

```

<div class="container">
    <h3 mat-dialog-title>Editar {{ENTIDAD}}</h3>

    <form class="mat-dialog-content" [formGroup]="familiaForm">

        <div class="form">
            <mat-form-field color="accent">
                <input matInput placeholder="{{ENTIDAD}}" formControlName="familia" name="familia"
                maxlength="100" required>
            </mat-form-field>
        </div>

        <div class="form">

```

```

        <mat-form-field color="accent">
            <textarea matInput placeholder="Observaciones" formControlName="observaciones"
name="Observaciones"></textarea>
        </mat-form-field>
    </div>

    <div class="form">
        <mat-form-field color="accent">
            <textarea matInput placeholder="Código Familia" formControlName="cod_familia"
name="Código Familia"></textarea>
        </mat-form-field>
    </div>

    <div mat-dialog-actions>
        <button mat-button id="onSubmit" type="submit" [disabled]="!familiaForm.valid"
(click)="confirmEdit()">Guardar</button>
        <button mat-button id="onNoClick" (click)="onNoClick()" tabindex="-1">Cancelar</button>
    </div>

</form>
</div>

```

TS

```

import { Component, OnInit, Inject } from '@angular/core';
import { FormGroup, FormControl, Validators } from '@angular/forms';
import { MatDialogRef, MAT_DIALOG_DATA } from '@angular/material/dialog';
import { MatSnackBar } from '@angular/material/snack-bar';
import { FamiliasService } from 'src/app/services/familias.service';
import { Familia } from 'src/app/shared/interfaces/familia';
import { CLOSE, ENTIDAD_FAMILIA, ERROR } from '../shared/messages';

@Component({
  selector: 'app-edit-familia',
  templateUrl: './edit-familia.component.html',
  styleUrls: ['./edit-familia.component.scss']
})
export class EditFamiliaComponent implements OnInit {

  familiaForm: FormGroup;
  ENTIDAD: String;

  constructor(
    public dialogRef: MatDialogRef<EditFamiliaComponent>,
    private snackBar: MatSnackBar,
    private servicioFamilias: FamiliasService,
    @Inject(MAT_DIALOG_DATA) public familia: Familia
  ) { }

  ngOnInit(): void {
    this.ENTIDAD = ENTIDAD_FAMILIA;
    this.familiaForm = new FormGroup({
      id_familia: new FormControl(this.familia.id_familia, Validators.required),
      familia: new FormControl(this.familia.familia, Validators.required),
      observaciones: new FormControl(this.familia.observaciones),
      cod_familia: new FormControl(this.familia.cod_familia)
    });
  }

  async confirmEdit(){

```

```

    if (this.familiaForm.valid) {
      const familiaForm = this.familiaForm.value;

      const RESPONSE = await this.servicioFamilias.editFamilia(familiaForm).toPromise();
      if (RESPONSE.ok) {
        this.snackBar.open(RESPONSE.message, CLOSE, { duration: 5000 });
        this.dialogRef.close({ ok: RESPONSE.ok, data: RESPONSE.data });
      } else { this.snackBar.open(ERROR, CLOSE, { duration: 5000 }); }
    } else { this.snackBar.open(ERROR, CLOSE, { duration: 5000 }); }
  }

  onNoClick() {
    this.dialogRef.close({ ok: false });
  }
}

```

Y para cambiar el INSERT, vamos a /src/app/familias/add-familia y editamos el html y ts de la siguiente manera:

HTML

```

<div class="container">
  <h3 mat-dialog-title>Añadir {{ENTIDAD}}</h3>

  <form class="mat-dialog-content" [formGroup]="familiaForm">

    <div class="form">
      <mat-form-field color="accent">
        <input matInput placeholder="{{ENTIDAD}}" formControlName="familia" name="familia"
maxlength="100" required>
      </mat-form-field>
    </div>

    <div class="form">
      <mat-form-field color="accent">
        <textarea matInput placeholder="Observaciones" formControlName="observaciones"
name="observaciones"></textarea>
      </mat-form-field>
    </div>

    <div class="form">
      <mat-form-field color="accent">
        <textarea matInput placeholder="Código Familia" formControlName="cod_familia"
name="Código Familia"></textarea>
      </mat-form-field>
    </div>

    <div mat-dialog-actions>
      <button mat-button id="onSubmit" type="submit" [disabled]="!familiaForm.valid"
(click)="confirmAdd()">Guardar</button>
      <button mat-button id="onNoClick" (click)="onNoClick()" tabindex="-1">Cancelar</button>
    </div>

  </form>
</div>

```

TS

```

import { Component, OnInit, Inject } from '@angular/core';
import { FormGroup, FormControl, Validators } from '@angular/forms';
import { MatDialogRef, MAT_DIALOG_DATA } from '@angular/material/dialog';
import { MatSnackBar } from '@angular/material/snack-bar';
import { FamiliasService } from 'src/app/services/familias.service';
import { Familia } from 'src/app/shared/interfaces/familia';
import { CLOSE, ENTIDAD_FAMILIA, ERROR } from '../shared/messages';

@Component({
  selector: 'app-edit-familia',
  templateUrl: './edit-familia.component.html',
  styleUrls: ['./edit-familia.component.scss']
})
export class EditFamiliaComponent implements OnInit {

  familiaForm: FormGroup;
  ENTIDAD: String;

  constructor(
    public dialogRef: MatDialogRef<EditFamiliaComponent>,
    private snackBar: MatSnackBar,
    private servicioFamilias: FamiliasService,
    @Inject(MAT_DIALOG_DATA) public familia: Familia
  ) { }

  ngOnInit(): void {
    this.ENTIDAD = ENTIDAD_FAMILIA;
    this.familiaForm = new FormGroup({
      id_familia: new FormControl(this.familia.id_familia, Validators.required),
      familia: new FormControl(this.familia.familia, Validators.required),
      observaciones: new FormControl(this.familia.observaciones),
      cod_familia: new FormControl(this.familia.cod_familia)
    });
  }

  async confirmEdit(){
    if (this.familiaForm.valid) {
      const familiaForm = this.familiaForm.value;

      const RESPONSE = await this.servicioFamilias.editFamilia(familiaForm).toPromise();
      if (RESPONSE.ok) {
        this.snackBar.open(RESPONSE.message, CLOSE, { duration: 5000 });
        this.dialogRef.close({ ok: RESPONSE.ok, data: RESPONSE.data });
      } else { this.snackBar.open(ERROR, CLOSE, { duration: 5000 }); }
    } else { this.snackBar.open(ERROR, CLOSE, { duration: 5000 }); }
  }

  onNoClick() {
    this.dialogRef.close({ ok: false });
  }
}

```

Y por ultimo, editamos las funciones de php que se encargan de interactuar con la base de datos en familia.php, situado en /api/private/apiClasses/, no confundir con el que se encuentra en /api/private/:

```
public function create($data) {
    $familia = $data['familia'];
    $observaciones = $data['observaciones'];
    $codFamilia = $data['cod_familia'];

    if (isset($familia)) {
        $sql = $this->conexion->prepare("INSERT INTO sgi_familias (familia, observaciones,
cod_familia)
        VALUES (:familia, :observaciones, :codFamilia)");
        $sql->bindParam(":familia", $familia, PDO::PARAM_STR);
        $sql->bindParam(":observaciones", $observaciones, PDO::PARAM_STR);
        $sql->bindParam(":codFamilia", $codFamilia, PDO::PARAM_STR);

        $resultado = $sql->execute();
        if ($resultado) {
            $this->status = true;
            $this->message = ADD_MODO_REUNION_OK;
        } else { $this->message = ADD_MODO_REUNION_KO; }

        $this->closeConnection();
    }
}

public function update($data) {
    $idFamilia = $data['id_familia'];
    $familia = $data['familia'];
    $observaciones = $data['observaciones'];
    $codFamilia = $data['cod_familia'];

    # Esto inserta NULL a observaciones si se encuentra vacío
    if ($observaciones == "")
        $observaciones = NULL;

    if (isset($familia)) {
        $sql = $this->conexion->prepare("UPDATE sgi_familias SET
        familia = :familia,
        observaciones = :observaciones,
        cod_familia = :codFamilia
        WHERE id_familia = :idFamilia");

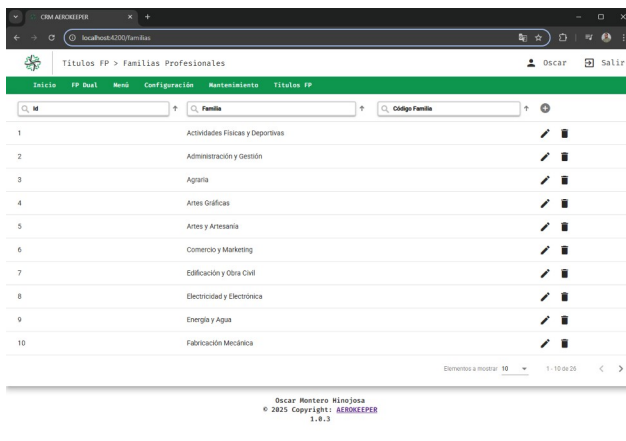
        $sql->bindParam(":familia", $familia, PDO::PARAM_STR);
        $sql->bindParam(":observaciones", $observaciones, PDO::PARAM_STR);
        $sql->bindParam(":codFamilia", $codFamilia, PDO::PARAM_STR);
        $sql->bindParam(":idFamilia", $idFamilia, PDO::PARAM_INT);

        $resultado = $sql->execute();
        if ($resultado) {
            $this->status = true;
            $this->message = EDIT_MODO_REUNION_OK;
        } else {
            $this->message = EDIT_MODO_REUNION_KO;
        }

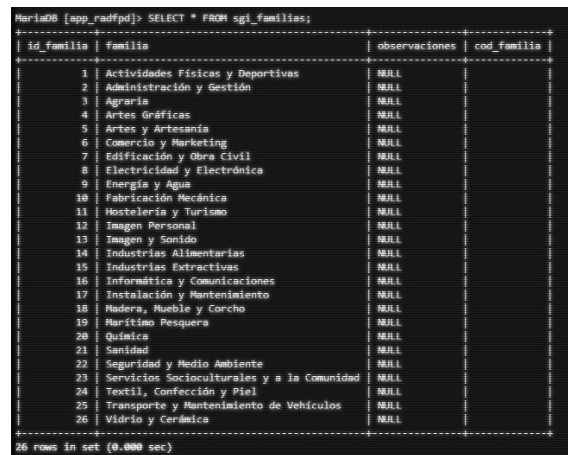
        $this->closeConnection();
    }
}
```


17/02/25

Resultado Estado inicial:



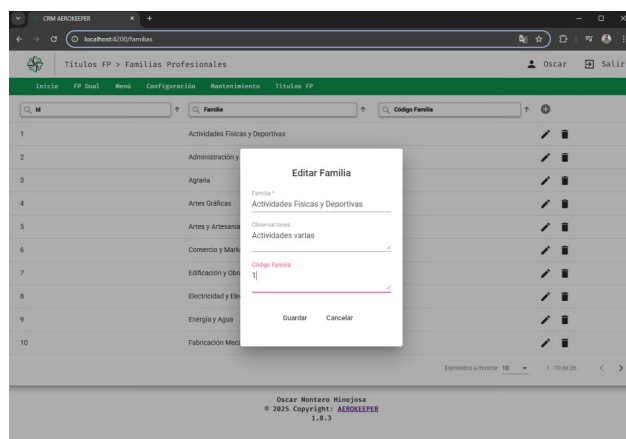
id_familia	Familia	observaciones	cod_familia
1	Actividades Físicas y Deportivas	NULL	
2	Administración y Gestión	NULL	
3	Agraria	NULL	
4	Artes Gráficas	NULL	
5	Artes y Artesanía	NULL	
6	Comercio y Marketing	NULL	
7	Edificación y Obra Civil	NULL	
8	Electricidad y Electrónica	NULL	
9	Energía y Agua	NULL	
10	Fabricación Mecánica	NULL	



```
MariaDB [app_radfpd]> SELECT * FROM sgl_familias;
```

id_familia	familia	observaciones	cod_familia
1	Actividades Físicas y Deportivas	NULL	
2	Administración y Gestión	NULL	
3	Agraria	NULL	
4	Artes Gráficas	NULL	
5	Artes y Artesanía	NULL	
6	Comercio y Marketing	NULL	
7	Edificación y Obra Civil	NULL	
8	Electricidad y Electrónica	NULL	
9	Energía y Agua	NULL	
10	Fabricación Mecánica	NULL	
11	Hostelería y Turismo	NULL	
12	Imagen Personal	NULL	
13	Imagen y Sonido	NULL	
14	Industrias Alimentarias	NULL	
15	Industrias Extractivas	NULL	
16	Informática y Comunicaciones	NULL	
17	Instalación y Mantenimiento	NULL	
18	Madera, Mueble y Corcho	NULL	
19	Marítimo Pesquera	NULL	
20	Química	NULL	
21	Sanidad	NULL	
22	Seguridad y Medio Ambiente	NULL	
23	Servicios Socioculturales y a la Comunidad	NULL	
24	Textil, Confección y Piel	NULL	
25	Transporte y Mantenimiento de Vehículos	NULL	
26	Vidrio y Cerámica	NULL	

Al realizar una edición:



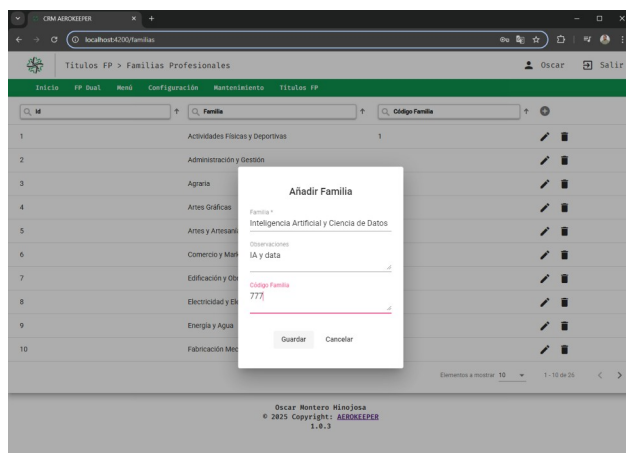
id_familia	Familia	observaciones	cod_familia
1	Actividades Físicas y Deportivas	Actividades varias	1
2	Administración y Gestión		
3	Agraria		



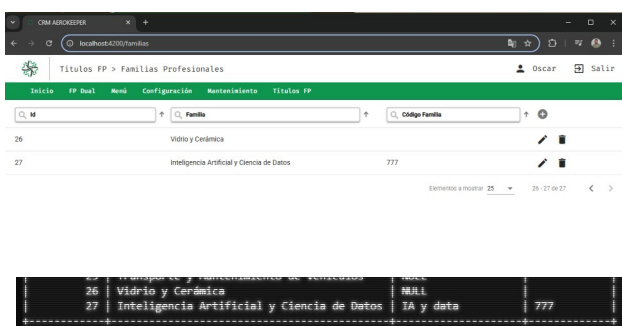
```
MariaDB [app_radfpd]> SELECT * FROM sgl_familias;
```

id_familia	familia	observaciones	cod_familia
1	Actividades Físicas y Deportivas	Actividades varias	1
2	Administración y Gestión	NULL	
3	Agraria	NULL	
4	Artes Gráficas	NULL	
5	Artes y Artesanía	NULL	

Al realizar una inserción:



id_familia	Familia	observaciones	cod_familia
1	Actividades Físicas y Deportivas		
2	Administración y Gestión		
3	Agraria		
4	Artes Gráficas		
5	Artes y Artesanía		
6	Comercio y Marketing		
7	Edificación y Obra Civil		
8	Electricidad y Electrónica		
9	Energía y Agua		
10	Fabricación Mecánica		



```
MariaDB [app_radfpd]> SELECT * FROM sgl_familias;
```

id_familia	familia	observaciones	cod_familia
26	Vidrio y Cerámica		
27	Inteligencia Artificial y Ciencia de Datos	IA y data	777