

## A4 - Projecte Agenda



```
ATTEMPT(S) LEFT: 3          RATING: 0
#KAYAKED!|: + <+?^(</>="$-M
#+=&_..... AZUMAS#$$KAB
<{"</WABBLE AKAS*$[@_%/
S{=[!,&%\]=. -.....+KAY
*^~KABUKISN~ AKER+_--{KA
KISHKAS$:?_& BIKIS=$//%:M
(:{"GALAXES ARKKAS^/<\; \>DUD REMOVED
-;([$-: ^_ ;N~ ?[N~<+_QABALA>KABUKIS
|+(=@[&*=?#[ S=(-: [" ,KNAC>5/7 correct
#*|~KNACKER[ KED[{" ;@_# , [>[!,&%\]
```

<b>1) INTRODUCCIÓ</b>	<b>3</b>
1.1) Creación de la BD y del usuario con permisos	3
<b>2) INSERIR EVENT</b>	<b>4</b>
2.1) Planteamiento de la tabla y su creación correspondiente	4
2.2) Planteamiento del procedimiento y su creación correspondiente	5
2.3) Funcionamiento del procedimiento	6
<b>3) MODIFICAR EVENT</b>	<b>7</b>
3.2) Planteamiento del procedimiento y su creación correspondiente	7
3.3) Funcionamiento del procedimiento	8
<b>4) ESBORRAR EVENT</b>	<b>8</b>
4.2) Planteamiento del procedimiento y su creación correspondiente	8
4.3) Funcionamiento del procedimiento	9
<b>5. AGENDA</b>	<b>9</b>
5.2) Planteamiento del procedimiento y su creación correspondiente	9
5.3) Funcionamiento del procedimiento	10
<b>6. INSERIR EVENT RECURRENT</b>	<b>10</b>
6.2) Planteamiento del procedimiento y su creación correspondiente	10
6.3) Funcionamiento del procedimiento	11
<b>7. ARXIVAT D'EVENTS</b>	<b>11</b>
7.2) Planteamiento del procedimiento y su creación correspondiente	11
7.3) Funcionamiento del procedimiento	12
<b>8. RECORDATORIS</b>	<b>13</b>
8.2) Planteamiento del procedimiento y su creación correspondiente	13

## 1) INTRODUCCIÓN

### 1.1) Creación de la BD y del usuario con permisos

Creamos el usuario 'a4\_valentin'.

```
valentin@ubuntupc: ~
postgres=# CREATE USER a4_valentin with PASSWORD '12345678';
CREATE ROLE
```

Creamos la BD 'a4\_agenda' con la opcion 'with owner' para asignar la propiedad al usuario creado anteriormente.

```
valentin@ubuntupc: ~
postgres=# CREATE DATABASE a4_agenda WITH OWNER a4_valentin;
CREATE DATABASE
postgres=#
```

Listamos las bases de datos existentes en el servidor y vemos que el dueño de la BD creada es el usuario correspondiente.

```
valentin@ubuntupc: ~
postgres=# \l
```

Nombre	Dueño	Codificación	Proveedor de locale	Collate	Ctype	configuración ICU	Reglas ICU:	Privilegios
a4_agenda	a4_valentin	UTF8	libc	es_ES.UTF-8	es_ES.UTF-8			
ecommerce	postgres	UTF8	libc	es_ES.UTF-8	es_ES.UTF-8			=Tc/postgres + postgres=CTc/postgres + admin=CTc/postgres
postgres	postgres	UTF8	libc	es_ES.UTF-8	es_ES.UTF-8			
template0	postgres	UTF8	libc	es_ES.UTF-8	es_ES.UTF-8			=c/postgres + postgres=CTc/postgres
template1	postgres	UTF8	libc	es_ES.UTF-8	es_ES.UTF-8			=c/postgres + postgres=CTc/postgres

```
(5 filas)
postgres=#
```

Finalmente, comprobamos que podamos conectar a la BD mediante el usuario y sus credenciales.

```
valentin@ubuntupc: ~
valentin@ubuntupc:~$ psql -U a4_valentin -d a4_agenda
Contraseña para usuario a4_valentin:
psql (16.3 (Ubuntu 16.3-1.pgdg22.04+1))
Digite «help» para obtener ayuda.
a4_agenda=>
```

## 2) INSERIR EVENT

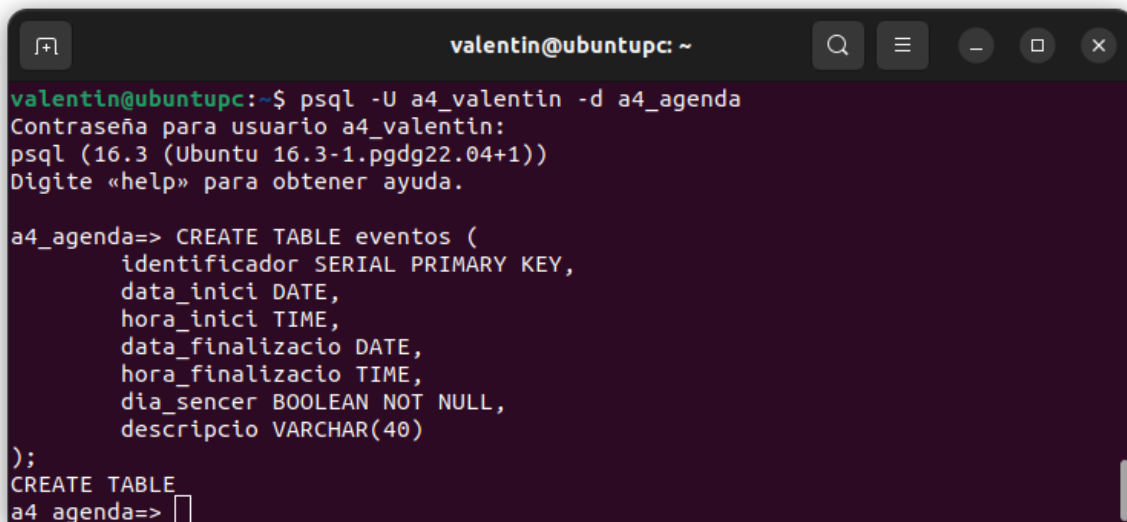
### 2.1) Planteamiento de la tabla y su creación correspondiente

Creemos la tabla **'eventos'** con los siguientes campos:

- Identificador: SERIAL
- Data d'inici de l'event: DATE
- Hora d'inici de l'event: TIME
- Data de finalització de l'event: DATE
- Hora de finalització de l'event: TIME
- Dia sencer: BOOLEAN

Query para la creación de la tabla en PostgreSQL:

```
CREATE TABLE eventos (  
    identificador SERIAL PRIMARY KEY,  
    data_inici DATE,  
    hora_inici TIME,  
    data_finalizacio DATE,  
    hora_finalizacio TIME,  
    dia_sencer BOOLEAN NOT NULL  
    descripcio VARCHAR(40)  
);
```



```
valentin@ubuntupc: ~  
valentin@ubuntupc:~$ psql -U a4_valentin -d a4_agenda  
Contraseña para usuario a4_valentin:  
psql (16.3 (Ubuntu 16.3-1.pgdg22.04+1))  
Digite «help» para obtener ayuda.  
  
a4_agenda=> CREATE TABLE eventos (  
    identificador SERIAL PRIMARY KEY,  
    data_inici DATE,  
    hora_inici TIME,  
    data_finalizacio DATE,  
    hora_finalizacio TIME,  
    dia_sencer BOOLEAN NOT NULL,  
    descripcio VARCHAR(40)  
);  
CREATE TABLE  
a4_agenda=>
```

## 2.2) Planteamiento del procedimiento y su creación correspondiente

```

CREATE OR REPLACE PROCEDURE insert_event(
    p_data_inici DATE,
    p_hora_inici TIME,
    p_descripcio VARCHAR(40),
    p_dia_sencer BOOLEAN,
    p_data_finalizacio DATE DEFAULT NULL,
    p_hora_finalizacio TIME DEFAULT NULL
)
LANGUAGE plpgsql
AS $$
BEGIN
    -- Verificar que la descripción no esté en blanco y 'TRIM()' para eliminar
    -- posibles espacios en blanco al principio y final de la cadena introducida.
    IF p_descripcio IS NULL OR TRIM(p_descripcio) = '' THEN
        RAISE EXCEPTION 'La descripció no pot estar en blanc.';
    END IF;

    -- Verificar si la fecha de inicio está en el pasado
    IF p_data_inici < CURRENT_DATE THEN
        RAISE EXCEPTION 'La data d'inici de l'event no pot estar en el passat.';
    END IF;

    -- Verificar si la fecha de finalización es anterior a la fecha de inicio
    IF p_data_finalizacio IS NOT NULL AND p_data_finalizacio < p_data_inici THEN
        RAISE EXCEPTION 'La data de fi de l'event no pot ser anterior a la data d'inici.';
    END IF;

    -- Verificar si la fecha de inicio y la fecha de finalización son iguales, en cuyo caso
    -- la hora de finalización debe ser posterior a la hora de inicio
    IF p_data_finalizacio = p_data_inici AND p_hora_finalizacio <= p_hora_inici THEN
        RAISE EXCEPTION 'Si la data d'inici i de fi de l'event són la mateixa, l'hora de fi
ha de ser posterior a l'hora d'inici.';
    END IF;

    -- Verificar si el evento es de día completo (dia_sencer)
    IF p_dia_sencer THEN
        -- Si es un evento de día completo, no deben proporcionarse horas de inicio o
        -- finalización
        IF p_hora_inici IS NOT NULL OR p_hora_finalizacio IS NOT NULL THEN
            RAISE EXCEPTION 'Si dia_sencer és true, només ha de haver data_inici i
data_finalizacio sense hora.';
        END IF;
    ELSE
        -- Si no es un evento de día completo, se espera que se proporcionen tanto fecha como
        -- hora de inicio
        IF p_hora_inici IS NULL THEN
            RAISE EXCEPTION 'Si dia_sencer és false, cal proporcionar hora d'inici.';
        END IF;
    END IF;
END;

```

```
        END IF;
    END IF;

    -- Insertar el evento si todas las condiciones son correctas
    INSERT INTO eventos (data_inici, hora_inici, data_finalizacio, hora_finalizacio,
dia_sencer, descripcio)
    VALUES (p_data_inici, p_hora_inici, p_data_finalizacio, p_hora_finalizacio, p_dia_sencer,
p_descripcio);
    -- Confirmar que la inserción fue exitosa
    RAISE NOTICE 'Inserció de l'event correctament.';
END;
$$;
```

## 2.3) Funcionamiento del procedimiento

Inserim un event nou.

```
a4_agenda=# CALL insert_event('2024-05-27', '14:00', 'chequeo del sistema', false, null, '14:00');
NOTICE:  Inserció de l'event correctament.
CALL
```

```
a4_agenda=# SELECT * FROM eventos;
 identificador | data_inici | hora_inici | data_finalizacio | hora_finalizacio | dia_sencer | descripcio
-----+-----+-----+-----+-----+-----+-----
          1 | 2024-05-27 | 14:00:00 |                  | 14:00:00         | f         | chequeo del sistema
(1 fila)
a4_agenda=#
```

### 3) MODIFICAR EVENT

#### 3.2) Planteamiento del procedimiento y su creación correspondiente

```

CREATE OR REPLACE PROCEDURE update_event (
    p_data_inici DATE,
    p_hora_inici TIME,
    p_nova_hora_finalizacio TIME DEFAULT NULL,
    p_nova_descripcio VARCHAR(40) DEFAULT NULL
)
LANGUAGE plpgsql
AS $$
BEGIN
    -- Verificar que el evento exista
    IF NOT EXISTS (
        SELECT 1
        FROM eventos
        WHERE data_inici = p_data_inici AND hora_inici = p_hora_inici
    ) THEN
        RAISE EXCEPTION 'L'event no existeix.';
    END IF;

    -- Verificar que al menos uno de los parámetros a actualizar no sea nulo
    IF p_nova_hora_finalizacio IS NULL AND p_nova_descripcio IS NULL THEN
        RAISE EXCEPTION 'Almenys un camp (hora de fi o descripció) ha de ser proporcionat
per a actualitzar.';
    END IF;

    -- Verificar que la nueva descripción no esté en blanco
    IF p_nova_descripcio IS NOT NULL AND TRIM(p_nova_descripcio) = '' THEN
        RAISE EXCEPTION 'La nova descripció no pot estar en blanc.';
    END IF;

    -- Verificar que la nueva hora de finalización sea posterior a la hora de inicio
    IF p_nova_hora_finalizacio IS NOT NULL THEN
        -- Obtener la hora de inicio del evento
        DECLARE v_hora_inici TIME;
        SELECT hora_inici INTO v_hora_inici
        FROM eventos
        WHERE data_inici = p_data_inici AND hora_inici = p_hora_inici;

        IF p_nova_hora_finalizacio <= v_hora_inici THEN
            RAISE EXCEPTION 'La nova hora de fi ha de ser posterior a l' hora d'inici.';
        END IF;
    END IF;

    -- Actualizar el evento
    UPDATE eventos
    SET
        hora_finalizacio = COALESCE(p_nova_hora_finalizacio, hora_finalizacio),
        descripcio = COALESCE(p_nova_descripcio, descripcio)
    WHERE data_inici = p_data_inici AND hora_inici = p_hora_inici;

```

```
-- Confirmar que la actualización fue exitosa
RAISE NOTICE 'Actualització de l'event correctament.';
END;
$$;
```

### 3.3) Funcionamiento del procedimiento

```
a4_agenda=# CALL update_event('2024-05-27', '14:00:00', '15:00:00', 'chequeo del sistema actualizado');
NOTICE: Actualització de l'event correctament.
CALL
a4_agenda=#
```

## 4) ESBORRAR EVENT

### 4.2) Planteamiento del procedimiento y su creación correspondiente

```
CREATE OR REPLACE PROCEDURE delete_event(
    p_data_inici DATE,
    p_hora_inici TIME
)
LANGUAGE plpgsql
AS $$
DECLARE
    v_event_count INT;
BEGIN
    -- Verificar que el evento exista
    SELECT COUNT(*) INTO v_event_count
    FROM eventos
    WHERE data_inici = p_data_inici AND hora_inici = p_hora_inici;

    IF v_event_count = 0 THEN
        RAISE EXCEPTION 'L'event no existeix.';
    END IF;

    -- Eliminar el evento
    DELETE FROM eventos
    WHERE data_inici = p_data_inici AND hora_inici = p_hora_inici;

    -- Confirmar que la eliminación fue exitosa
    RAISE NOTICE 'Esborrat de l'event correctament.';
END;
$$;
```



### 4.3) Funcionamiento del procedimiento

```
a4_agenda=# select * from eventos;
identificador | data_inici | hora_inici | data_finalizacio | hora_finalizacio | dia_sencer | descripcio
-----+-----+-----+-----+-----+-----+-----
1 | 2024-05-27 | 14:00:00 | | 15:00:00 | f | chequeo del sistema actualizado
(1 fila)

a4_agenda=# call delete_event('2024-05-27', '14:00:00');
NOTICE: Esborrat de l'event correctament.
CALL
a4_agenda=# select * from eventos;
identificador | data_inici | hora_inici | data_finalizacio | hora_finalizacio | dia_sencer | descripcio
-----+-----+-----+-----+-----+-----+-----
(0 filas)

a4_agenda=#
```

## 5. AGENDA

### 5.2) Planteamiento del procedimiento y su creación correspondiente

```
CREATE OR REPLACE FUNCTION agenda(
    p_opcio CHAR DEFAULT 'D',
    p_data DATE DEFAULT CURRENT_DATE
)
RETURNS TABLE(
    data_inici DATE,
    hora_inici TIME,
    descripcio VARCHAR(40),
    data_finalizacio DATE,
    hora_finalizacio TIME,
    dia_sencer BOOLEAN
)
LANGUAGE plpgsql
AS $$
BEGIN
    IF p_opcio = 'D' THEN
        RETURN QUERY
        SELECT e.data_inici, e.hora_inici, e.descripcio, e.data_finalizacio,
e.hora_finalizacio, e.dia_sencer
        FROM eventos e
        WHERE e.data_inici = p_data;

    ELSIF p_opcio = 'S' THEN
        RETURN QUERY
        SELECT e.data_inici, e.hora_inici, e.descripcio, e.data_finalizacio,
e.hora_finalizacio, e.dia_sencer
        FROM eventos e
        WHERE DATE_TRUNC('week', e.data_inici) = DATE_TRUNC('week', p_data);

    ELSIF p_opcio = 'M' THEN
        RETURN QUERY
        SELECT e.data_inici, e.hora_inici, e.descripcio, e.data_finalizacio,
e.hora_finalizacio, e.dia_sencer
        FROM eventos e
        WHERE DATE_TRUNC('month', e.data_inici) = DATE_TRUNC('month', p_data);
```

```

ELSIF p_opcio = 'A' THEN
    RETURN QUERY
    SELECT e.data_inici, e.hora_inici, e.descripcio, e.data_finalizacio,
e.hora_finalizacio, e.dia_sencer
    FROM eventos e
    WHERE e.data_inici >= p_data;

ELSE
    RAISE EXCEPTION 'Opció no vàlida. Els valors possibles són: D, S, M, A.';
END IF;
END;
$$;

```

### 5.3) Funcionamiento del procedimiento

```

a4_agenda=# SELECT * FROM agenda('M', '2024-07-01');
 data_inici | hora_inici |      descripcio      | data_finalizacio | hora_finalizacio | dia_sencer
-----+-----+-----+-----+-----+-----
 2024-07-02 | 09:00:00  | Cita con el doctor   | 2024-07-02      | 10:00:00        | f
 2024-07-03 |           | Conferencia internacional | 2024-07-05      |                 | t
 2024-07-06 | 14:00:00  | Almuerzo con cliente | 2024-07-06      | 15:00:00        | f
 2024-07-07 | 18:00:00  | Revisión equipo de desarrollo | 2024-07-07      | 20:00:00        | f
(4 filas)

a4_agenda=# SELECT * FROM agenda('D', '2024-07-02');
 data_inici | hora_inici |      descripcio      | data_finalizacio | hora_finalizacio | dia_sencer
-----+-----+-----+-----+-----+-----
 2024-07-02 | 09:00:00  | Cita con el doctor   | 2024-07-02      | 10:00:00        | f
(1 fila)

a4_agenda=#

```

## 6. INSERTAR EVENT RECURRENT

### 6.2) Planteamiento del procedimiento y su creación correspondiente

```

CREATE OR REPLACE PROCEDURE insert_recurring_event(
    p_data_inici DATE,
    p_hora_inici TIME,
    p_descripcio VARCHAR(40),
    p_periodicitat INTEGER,
    p_data_finalizacio DATE DEFAULT NULL,
    p_numero_repeticions INTEGER DEFAULT NULL
)
LANGUAGE plpgsql
AS $$
DECLARE
    v_data_actual DATE := p_data_inici;
    v_repeticions INTEGER := 0;
BEGIN
    -- Comprobar que solo se haya proporcionado uno de los dos parámetros
    IF (p_data_finalizacio IS NOT NULL AND p_numero_repeticions IS NOT NULL) OR
        (p_data_finalizacio IS NULL AND p_numero_repeticions IS NULL) THEN
        RAISE EXCEPTION 'Cal especificar només una de les dues opcions: data fi o número de
repeticions.';
    END IF;

```

```

-- Insertar eventos recurrentes
LOOP
    -- Insertar el evento actual
    CALL insert_event(v_data_actual, p_hora_inici, p_descripcio, false, null, null);
    v_repeticions := v_repeticions + 1;

    -- Comprobar si se ha llegado a la fecha final o al número de repeticiones
    IF (p_data_finalizacio IS NOT NULL AND v_data_actual >= p_data_finalizacio) OR
        (p_numero_repeticions IS NOT NULL AND v_repeticions >= p_numero_repeticions) THEN
        EXIT;
    END IF;

    -- Incrementar la fecha actual según la periodicitat
    v_data_actual := v_data_actual + p_periodicitat;
END LOOP;

-- Mostrar mensaje de finalización
RAISE NOTICE 'Recurrença finalitzada, arribat a la data: %.', v_data_actual;

END;
$$;

```

## 6.3) Funcionamiento del procedimiento

```

a4_agenda=# CALL insert_recurring_event('2024-06-16', '10:30', 'dentista', 14, null, 5);
NOTICE: Inserció de l'event correctament.
NOTICE: Inserció de l'event correctament.
NOTICE: Inserció de l'event correctament.
NOTICE: Inserció de l'event correctament.
NOTICE: Inserció de l'event correctament.
NOTICE: Recurrença finalitzada, arribat a la data: 2024-08-11.
CALL
a4_agenda=#

```

## 7. ARXIVAT D'EVENTS

### 7.2) Planteamiento del procedimiento y su creación correspondiente

```

CREATE TABLE eventos_archivados (
    identificador SERIAL PRIMARY KEY,
    data_inici DATE,
    hora_inici TIME,
    data_finalizacio DATE,
    hora_finalizacio TIME,
    dia_sencer BOOLEAN,
    descripcio VARCHAR(40)
);

CREATE OR REPLACE PROCEDURE archivar_eventos_pasados()
LANGUAGE plpgsql
AS $$
BEGIN
    -- Seleccionar los eventos pasados

```

```
INSERT INTO eventos_archivados (data_inici, hora_inici, data_finalizacio,
hora_finalizacio, dia_sencer, descripcio)
SELECT data_inici, hora_inici, data_finalizacio, hora_finalizacio, dia_sencer,
descripcio
FROM eventos
WHERE data_inici < CURRENT_DATE;

-- Eliminar los eventos archivados de la tabla original
DELETE FROM eventos
WHERE data_inici < CURRENT_DATE;

RAISE NOTICE 'Events arxivats amb èxit.';
END;
$$;
```

### 7.3) Funcionamiento del procedimiento

```
a4_agenda=# CREATE TABLE eventos_archivados (
    identificador SERIAL PRIMARY KEY,
    data_inici DATE,
    hora_inici TIME,
    data_finalizacio DATE,
    hora_finalizacio TIME,
    dia_sencer BOOLEAN,
    descripcio VARCHAR(40)
);
CREATE TABLE
a4_agenda=# CREATE OR REPLACE PROCEDURE archivar_eventos_pasados()
LANGUAGE plpgsql
AS $$
BEGIN
    -- Seleccionar los eventos pasados
    INSERT INTO eventos_archivados (data_inici, hora_inici, data_finalizacio, hora_finalizacio, dia_sencer, descripcio)
    SELECT data_inici, hora_inici, data_finalizacio, hora_finalizacio, dia_sencer, descripcio
    FROM eventos
    WHERE data_inici < CURRENT_DATE;

    -- Eliminar los eventos archivados de la tabla original
    DELETE FROM eventos
    WHERE data_inici < CURRENT_DATE;

    RAISE NOTICE 'Events arxivats amb èxit.';
END;
$$;
CREATE PROCEDURE
a4_agenda=#
```

## 8. RECORDATORIS

### 8.2) Planteamiento del procedimiento y su creación correspondiente

```

CREATE TABLE recordatorios (
    id SERIAL PRIMARY KEY,
    minutos INTEGER,
    text_minutos VARCHAR(20)
);

INSERT INTO recordatorios (minutos, text_minutos)
VALUES (30, '30 minuts abans'),
       (60, '1 hora abans'),
       (240, '4 hores abans'),
       (1440, '1 dia abans'),
       (2880, '2 dies abans');

CREATE TABLE recordatorios_eventos (
    id_evento INTEGER REFERENCES eventos(identificador),
    id_recordatorio INTEGER REFERENCES recordatorios(id),
    fecha_recordatorio TIMESTAMP
);

CREATE OR REPLACE FUNCTION insertar_recordatorio_evento(
    p_id_evento INTEGER,
    p_id_recordatorio INTEGER
)
RETURNS VOID
LANGUAGE plpgsql
AS $$
BEGIN
    INSERT INTO recordatorios_eventos (id_evento, id_recordatorio, fecha_recordatorio)
    VALUES (p_id_evento, p_id_recordatorio, NULL);
END;
$$;

CREATE OR REPLACE PROCEDURE insert_event(
    p_data_inici DATE,
    p_hora_inici TIME,
    p_descripcio VARCHAR(40),
    p_dia_sencer BOOLEAN,
    p_data_finalizacio DATE DEFAULT NULL,
    p_hora_finalizacio TIME DEFAULT NULL
)
LANGUAGE plpgsql
AS $$
DECLARE
    v_id_evento INTEGER;
BEGIN
    -- Insertar el evento
    INSERT INTO eventos (data_inici, hora_inici, data_finalizacio, hora_finalizacio,
    dia_sencer, descripcio)

```

```
VALUES (p_data_inici, p_hora_inici, p_data_finalizacio, p_hora_finalizacio,
p_dia_sencer, p_descripcio)
RETURNING identificador INTO v_id_evento;

-- Insertar recordatorios predeterminados
INSERT INTO recordatorios_eventos (id_evento, id_recordatorio, fecha_recordatorio)
SELECT v_id_evento, id, NULL
FROM recordatorios
WHERE minutos IN (60, 1440);

-- Mostrar mensaje de inserción exitosa
RAISE NOTICE 'Inserció de l'event correctament.';

END;
$;$
```