Aquí te muestro cómo podrías proceder, asumiendo que quieres evitar duplicados basados en `site_id` y `document_name`:

1. **Agregar una restricción de unicidad a tu tabla** (si es que aún no existe y es aplicable a tu cas

sql Copy code

```
ALTER TABLE public.site_documents ADD CONSTRAINT site_document_unique UNIQUE
```

1. **Usar `ON CONFLICT` en tu consulta de inserción** para actualizar el registro existente en caso de conflicto:

sql Copy code

```
INSERT INTO public.site_documents (site_id, document_name, document_type, ren
VALUES (1, 'Documento1', 'Tipo1', '2023-01-01', nextval('site_documents_docum
ON CONFLICT (site_id, document_name) DO UPDATE SET
document_type = EXCLUDED.document_type,
renovation_date = EXCLUDED.renovation_date;
```

En este ejemplo, si intentas insertar un documento que tiene el mismo `site_id` y `document_name` que un registro existente, PostgreSQL actualizará el `document_type` y `renovation_date` del registro existente en lugar de insertar un nuevo registro. `EXCLUDED` s refiere a los valores que habrían sido insertados pero fueron excluidos debido al conflicto.

Es importante mencionar que el uso de `nextval` para `document_id` en este contexto podr no ser lo ideal si quieres reutilizar el ID para documentos actualizados. Si el `document_id` de ser único y consecutivo para cada nuevo documento, independientemente de las actualizaciones, entonces está bien. Si no, podrías necesitar revisar cómo manejas los IDs par que se ajusten a tus necesidades específicas.