perplexity

HOW MAKE ID AUTO INC PGSQL

To create an auto-incrementing ID column in PostgreSQL, you can use the `SERIAL` data type or the `GENERATED AS IDENTITY` feature. The `SERIAL` method is simpler, but `GENERATED AS IDENTITY` is recommended for newer versions (PostgreSQL 10+) because it avoids potential issues with accidental value overrides[1].

Here are the approaches:

1. Using SERIAL (Older PostgreSQL versions)
When creating a table, define the ID column as `SERIAL PRIMARY KEY`[2]:

```
CREATE TABLE table_name (
    id SERIAL PRIMARY KEY,
    other_column data_type
);
```

`SERIAL` is a shorthand notation that tells PostgreSQL to create an auto-incrementing, unique identifier for the specified column[2]. PostgreSQL automatically creates a sequence object and sets the default value of the column to the next value from the sequence[1].

2. Using GENERATED AS IDENTITY (PostgreSQL 10+)
When creating a table, define the ID column using `GENERATED ALWAYS AS IDENTITY` or `GENERATED BY DEFAULT AS IDENTITY`[1]:

```
CREATE TABLE table_name (
    id INT GENERATED ALWAYS AS IDENTITY PRIMARY KEY,
    other_column data_type
);
```

or

```
CREATE TABLE table_name (
    id INT GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,
    other_column data_type
);
```

- `GENERATED ALWAYS AS IDENTITY` ensures that the column value is always generated by the sequence and cannot be overridden by user input[1].
- `GENERATED BY DEFAULT AS IDENTITY` allows you to explicitly insert a value into the identity column, but if you don't, the sequence will generate the value[1].

3. Adding an auto-incrementing ID to an existing table

PostgreSQL 10+

```
ALTER TABLE table_name ADD COLUMN id INT GENERATED ALWAYS AS IDENTITY PRIMARY KEY;
```

Older versions of PostgreSQL

```
ALTER TABLE table_name ADD COLUMN id SERIAL PRIMARY KEY;
```

If the table already contains data, you might encounter an error if the new ID column is defined as `NOT NULL` without populating it first. In this case, you can add the column, populate it with values from a sequence, and then add the primary key constraint[1].

If you need more control over the sequence (e.g., custom starting value or increment), you can create a custom sequence and use it for the ID column[2]:

```
CREATE SEQUENCE books_sequence START 100 INCREMENT 2;

CREATE TABLE books (
    id INT PRIMARY KEY DEFAULT nextval('books_sequence'),
    title VARCHAR(100) NOT NULL,
    primary_author VARCHAR(100)
);
```

In this case, every time you insert a new record into the `books` table without specifying the `id`, it will use the next value from the `books_sequence`[2].

⁂

1. https://stackoverflow.com/questions/2944499/how-to-add-an-auto-incrementing-primary-key-to-an-existing-table-in-postgresql
2. https://www.atlassian.com/data/admin/how-to-define-an-auto-increment-primary-key-in-postgresql