Managing Created Tables: Takeaways

Syntax

- Renaming a table:

```
import psycopg2
conn = psycopg2.connect("dbname=dq user=dq")
cur = conn.cursor()
cur.execute('ALTER TABLE old_table RENAME TO new_table')
conn.commit()
```

- Dropping a column:

```
import psycopg2
conn = psycopg2.connect("dbname=dq user=dq")
cur = conn.cursor()
cur.execute('ALTER TABLE example_table DROP COLUMN redundant_column')
conn.commit()
```

- Mapping a value to the data type:

```
import psycopg2
from psycopg2 import extensions as ext
conn = psycopg2.connect("dbname=dq user=dq")
cur = conn.cursor()
cur.execute('SELECT id FROM ign_reviews')
id_column = cur.description[0]
print(id_column.type_code in ext.LONGINTEGER.values) ## prints False
```

- Changing a columns data type:

```
import psycopg2
conn = psycopg2.connect("dbname=dq user=dq")
cur = conn.cursor()
 # Assume `other_type` is of type `INTEGER`.
cur.execute('ALTER TABLE example_table ALTER COLUMN other_type TYPE BIGINT')
conn.commit()
```

- Renaming a column:

```
import psycopg2

conn = psycopg2.connect("dbname=dq user=dq")

cur = conn.cursor()

cur.execute('ALTER TABLE example_table RENAME COLUMN bad_name TO relevant_name')

conn.commit()
```

- Adding a column:

```
import psycopg2

conn = psycopg2.connect("dbname=dq user=dq")

cur = conn.cursor()

cur.execute('ALTER TABLE example_table ADD COLUMN id INTEGER PRIMARY KEY')

conn.commit()
```

- Adding a column with a default value:

```
ALTER TABLE ign_reviews ADD COLUMN release_date DATE DEFAULT 01-01-1991
```

- Adding or updating entries in a column:

```
UPDATE ign_reviews SET editors_choice = 'F' WHERE id > 5000
```

- Creating a date type from text:

```
to_date('01-01-1991', 'DD-MM-YYYY')
```

Concepts

- It's always a good idea to remove redundant columns as tables taking unnecessary disk space can cause queries to be slower.
- You can not change the data type of a column to another that it is not compatible with.
- Adding a column will result in that column containing null entries.
- We can concatenate strings using `||` . `||` is similar to `+` in Python, and it is part of Postgres' built-in functions that can be used to create new entries from a combination or already declared columns.

Resources

- Data Type Formatting Functions
- Documentation for Alter Table