Creating Table

Print to PDF ▶

Before getting into action with respect to basic DML and queries or CRUD operations, we need to prepare tables.

At this time we have not covered DDL yet. All database operations related to managing tables come under DDL.

For now, let's just create the table by copy pasting below `CREATE TABLE` statement. We will get into concepts as part of the subsequent sections.

- Connect to the database.
- Create the table.

```
%load_ext sql
```

```
%env DATABASE_URL=postgresql://itversity_sms_user:sms_password@localhost:5432/itversity_sms_db
```

```
env:
DATABASE_URL=postgresql://itversity_sms_user:sms_password@localhost:5432/itversity_sms_db
```

```
%%sql

SELECT * FROM information_schema.tables
WHERE table_catalog = 'itversity_sms_db' AND table_schema = 'public'
LIMIT 10
```

```
2 rows affected.
```

table_catalog	table_schema	table_name	table_type	self_referencing_column_name	referen
itversity_sms_db	public	courses	BASE TABLE	None	
itversity_sms_db	public	users	BASE TABLE	None	

```
%%sql

DROP TABLE IF EXISTS users;
```

```
 * postgresql://itversity_sms_user:***@localhost:5432/itversity_sms_db
Done.
```

```
[]
```

```
%%sql

SELECT * FROM information_schema.tables
WHERE table_catalog = 'itversity_sms_db' AND table_schema = 'public'
LIMIT 10
```

```
 * postgresql://itversity_sms_user:***@localhost:5432/itversity_sms_db
1 rows affected.
```

table_catalog	table_schema	table_name	table_type	self_referencing_column_name	referen
itversity_sms_db	public	courses	BASE TABLE	None	

```
%%sql

CREATE TABLE users (
    user_id SERIAL PRIMARY KEY,
    user_first_name VARCHAR(30) NOT NULL,
    user_last_name VARCHAR(30) NOT NULL,
    user_email_id VARCHAR(50) NOT NULL,
    user_email_validated BOOLEAN DEFAULT FALSE,
    user_password VARCHAR(200),
    user_role VARCHAR(1) NOT NULL DEFAULT 'U', --U and A
    is_active BOOLEAN DEFAULT FALSE,
    create_ts TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    last_updated_ts TIMESTAMP DEFAULT CURRENT_TIMESTAMP
)
```

```
 * postgresql://itversity_sms_user:***@localhost:5432/itversity_sms_db
Done.
```

```
[]
```

- Let us validate the objects that are created in the underlying database. We can either run query against **information_schema** or use Database Explorer in **SQL Workbench** or even `psql`.

```
%%sql

SELECT * FROM information_schema.tables
WHERE table_catalog = 'itversity_sms_db' AND table_schema = 'public'
LIMIT 10
```

```
 * postgresql://itversity_sms_user:***@localhost:5432/itversity_sms_db
1 rows affected.
```

table_catalog	table_schema	table_name	table_type	self_referencing_column_name	referen
itversity_sms_db	public	users	BASE TABLE	None	

```
%%sql

SELECT * FROM information_schema.columns
WHERE table_name = 'users'
LIMIT 10
```

```
 * postgresql://itversity_sms_user:***@localhost:5432/itversity_sms_db
10 rows affected.
```

table_catalog	table_schema	table_name	column_name	ordinal_position	
itversity_sms_db	public	users	user_id	1	nextval('user
itversity_sms_db	public	users	user_first_name	2	
itversity_sms_db	public	users	user_last_name	3	
itversity_sms_db	public	users	user_email_id	4	
itversity_sms_db	public	users	user_email_validated	5	
itversity_sms_db	public	users	user_password	6	
itversity_sms_db	public	users	user_role	7	
itversity_sms_db	public	users	is_active	8	
itversity_sms_db	public	users	create_ts	9	(
itversity_sms_db	public	users	last_updated_ts	10	(

```
%sql SELECT * FROM users
```

```
 * postgresql://itversity_sms_user:***@localhost:5432/itversity_sms_db
0 rows affected.
```

user_id	user_first_name	user_last_name	user_email_id	user_email_validated	user_password	u

By Durga Gadiraju