```
---
title: "test"
format:
    pdf:
        toc: true
        number-section: true
        colorlinks: true
jupyter: python3
---
```

```
SyntaxError: invalid syntax (1533296891.py, line 1)
```

```
import psycopg2
from psycopg2 import sql
from psycopg2.extensions import ISOLATION_LEVEL_AUTOCOMMIT # <-- ADD THIS LINE
```

```
# run these commands inside
!docker run --name pg_local -p 5432:5432 -e POSTGRES_USER=sde -e POSTGRES_PASSWORD=passwor
```

```
!pgcli -h localhost -p 5432 -U sde scd2
# password is password
```

```
# make sure you have docker running
# and your postgresql container running
!docker ps
```

```
CONTAINER ID   IMAGE           COMMAND                  CREATED          STATUS          POR
e96549e4bce0   postgres:12.2   "docker-entrypoint.s…"   46 minutes ago   Up 46 minutes   0.0
```

```
dbname='scd2'
user = 'sde'
host='localhost:5432'
password = 'password'
```

```
connection = psycopg2.connect(f"dbname={dbname} user={user} password={password}")
```

```
connection.autocommit = True
```

```
cur = connection.cursor()
```

```
print('PostgreSQL database version:')
cur.execute('SELECT version()')
```

```
PostgreSQL database version:
```

```
db_version = cur.fetchone()
```

```
print(db_version)
```

```
('PostgreSQL 12.2 (Debian 12.2-2.pgdg100+1) on x86_64-pc-linux-gnu, compiled by gcc (Debian 8
```

```
command = """
CREATE DATABASE warehouse;
"""
```

```
cur.execute(command)
```

```
# create user table and fill it
user_table_command = """
DROP TABLE IF EXISTS user_dim;
CREATE TABLE user_dim (
    user_key BIGINT,
    user_id VARCHAR(40),
    first_name VARCHAR(10),
    last_name VARCHAR(10),
    address VARCHAR(100),
    zipcode VARCHAR(10),
    created_datetime TIMESTAMP,
    updated_datetime TIMESTAMP,
    row_effective_datetime TIMESTAMP,
    row_expiration_datetime TIMESTAMP,
    current_row_indicator VARCHAR(10)
);
INSERT INTO user_dim (
        user_key,
        user_id,
        first_name,
        last_name,
        address,
        zipcode,
        created_datetime,
        updated_datetime,
        row_effective_datetime,
        row_expiration_datetime,
        current_row_indicator
    )
VALUES (
        1000,
        'b0cc9fde-a29a-498e-824f-e52399991beb',
        'john',
        'doe',
        'world',
        10027,
        '2020-01-01 10:00:00',
        '2020-01-01 10:00:00',
        ' 2020-01-01 10:00:00',
        '2021-01-01 17:59:59',
        'expired'
    ),
    (
        1200,
        'b0cc9fde-a29a-498e-824f-e52399991beb',
        'john',
        'doe',
        'world',
        10012,
        '2020-01-01 10:00:00',
        '2021-01-01 18:00:00',
        '2021-01-01 18:00:00',
        '9999-12-31 00:00:00',
        'current'
    );
"""
```

```
cur.execute(user_table_command)
```

```
# now we check our work
select_user_table = """
select *
from user_dim
"""
```

```
cur.execute(select_user_table)
```

```
records = cur.fetchall()
```

```
for row in records:
    print(row)
```

```
(1000, 'b0cc9fde-a29a-498e-824f-e52399991beb', 'john', 'doe', 'world', '10027', datetime.date
(1200, 'b0cc9fde-a29a-498e-824f-e52399991beb', 'john', 'doe', 'world', '10012', datetime.date
```