




Guía: Liberar conexiones al Pool con `psycopg_pool`

Introducción

En versiones recientes de `psycopg_pool`, **ya no es necesario liberar manualmente las conexiones** si se usa `with`, ya que esto se hace automáticamente mediante el *Resource Manager* del contexto.

Este comportamiento hace que la gestión de conexiones sea más eficiente y menos propensa a errores. Sin embargo, en esta lección también veremos cómo se hacía tradicionalmente mediante métodos como `putconn()`, únicamente como referencia histórica o en casos donde **no se use `with`**.

Liberación automática con `with` (recomendado)


 Archivo: `capa_datos_persona/conexion.py`

Cuando usas este método:

```
@classmethod
def obtener_conexion(cls):
    return cls._pool.connection()
```

Entonces puedes obtener y liberar conexiones así:

```
with Conexion.obtener_conexion() as conexion:
    # Realiza operaciones con la base de datos
    log.debug(f'Conexión utilizada: id={id(conexion)}')
```

 Al finalizar el bloque `with`, la conexión se **regresa automáticamente al pool**. Esto hace innecesario llamar a `putconn()` o cerrar conexiones manualmente.

Liberación manual con `putconn()` (solo para referencia)

Si decides usar el método `getconn()` para obtener una conexión **fuera de un bloque `with`**, entonces sí necesitas liberarla con `putconn()`:

```
@classmethod
def obtener_conexion(cls):
    return cls._pool.getconn()

@classmethod
def liberar_conexion(cls, conexion):
    cls._pool.putconn(conexion)
    log.debug(f'Regresamos la conexión al pool: id={id(conexion)}')
```

 Ejemplo de uso:

```
conexion = Conexion.obtener_conexion()
# ... operaciones
Conexion.liberar_conexion(conexion)
```

Cierre total del pool (opcional al finalizar la app)

Puedes cerrar completamente el pool para liberar todos los recursos con:

```
@classmethod
def cerrar_conexiones(cls):
    cls._pool.close()
    log.debug('Pool de conexiones cerrado correctamente.')
```

Esto es útil en pruebas o al terminar una aplicación.

Prueba práctica: usando `with`

```
for i in range(6): # Se pueden solicitar más de 5 conexiones secuenciales
    with Conexion.obtener_conexion() as conexion:
        log.debug(f'Conexión #{i + 1} usada correctamente: id={id(conexion)}')
```

Gracias a que cada conexión se libera automáticamente, **el pool no se agota**.

Prueba práctica: sin liberar (con error esperado)




```
conexiones = []
for i in range(6): # Solo 5 disponibles
    conexion = Conexion.obtener_conexion()
    conexiones.append(conexion)
```

Esto arrojará:

```
psycopg_pool.PoolTimeout: connection pool exhausted
```

✖ A menos que liberes manualmente con `putconn()`.

Conclusión

-  Usa `with + connection()` → **no necesitas liberar nada**.
 -  Usa `getconn() + putconn()` solo si lo haces manualmente.
 -  Usa `close()` si quieres terminar el pool por completo.
-

✨ Sigue adelante con tu aprendizaje  , ¡el esfuerzo vale la pena!

¡Saludos! 

Ing. Marcela Gamiño e Ing. Ubaldo Acosta

Fundadores de [GlobalMentoring.com.mx](https://www.globalmentoring.com.mx)