GO QUICK REFERENCE GUIDE — database/sql

sql.Open: Create a sql.DB connection pool

```go
db, err := sql.Open("DRIVER_NAME", "CONNECTION_STRING")
if err != nil {
    log.Fatal(err)
}
defer db.Close()
// Use Ping() to create a connection and check for any errors.
if err = db.Ping(); err != nil {
    log.Fatal(err)
}
```

db.Exec: Execute a SQL statement

```go
result, err := db.Exec("INSERT INTO customers (name) VALUES (?)", "Alice")
if err != nil {
    log.Fatal(err)
}
// Note: The LastInsertID() method is not supported by PostgreSQL. Use the
// db.QueryRow() method with a "RETURNING" clause instead.
id, err := result.LastInsertId()
if err != nil {
    log.Fatal(err)
}
affected, err := result.RowsAffected()
if err != nil {
    log.Fatal(err)
}
```

db.QueryRow: Fetch a single row

```go
var name string
err := db.QueryRow("SELECT name FROM customers WHERE id = ?", 1).Scan(&name)
if err == sql.ErrNoRows {
    log.Fatal("no rows returned")
} else if err != nil {
    log.Fatal(err)
}
```

db.Query: Fetch multiple rows

```go
rows, err := db.Query("SELECT name FROM customers LIMIT 10")
if err != nil {
    log.Fatal(err)
}
defer rows.Close()

names := []string{}
for rows.Next() {
    var name string
    err := rows.Scan(&name)
    if err != nil {
        log.Fatal(err)
    }
    names = append(names, name)
}

if err = rows.Err(); err != nil {
    log.Fatal(err)
}
```

sql.Tx: Execute multiple statements in one transaction

```go
// All statements in a transaction use the same database connection. Either all
// statements are executed successfully, or none at all.
tx, err := db.Begin()
if err != nil {
    log.Fatal(err)
}

_, err := tx.Exec("INSERT INTO customers ...")
if err != nil {
    tx.Rollback()
    log.Fatal(err)
}

// ...

err = tx.Commit()
```

PostgreSQL: Use $N notation for placeholder parameters instead of the ? character.