

Go and Data Persistence

Why Data Persistence

- Preserve Data if Server crashes
- Scalability of applications
- High throughput and availability

Types of Databases

- Sql (MySQL, Postgres)
- NoSql (Mongo, DynamoDb)
- Timeseries
- Graph Db (can be implemented with Mongo)

How do we connect to a database ?

- Need a “driver” to connect to our database
- Can write queries directly in the native programming language
- Can use ORM (Object Relational Mapper)
- We will write our own queries

Example Create Statement for Postgres

- `sqlStatement := `INSERT INTO table1 (field1, field2, field3, field4) VALUES ($1, $2, $3, $4)``
- Inserts into “table1” that db is connected to a new row with values for field1, field2, field3 and field4 (the columns)
- Need to store those fields in a struct and then reference them at invocation
- Need to allow for error handling

Example Select Statement for Postgres

- `sqlStatement := "SELECT field1, field2, field3, field4 FROM table1"`
- Returns field1, fields, field3 and field4 from table1
- Can be all the fields, i.e. the entire row
- Can be subset of fields
- Depends on the needs of the application

Example Update Statement for Postgres

- `sqlStatement := `UPDATE books SET title = $1, author = $2, isbn = $3, price = $4 WHERE id = $5``
- Updates into “table1” that db is connected to values for object whose id is value is represented by \$5
- Fields can be all the columns for that row or a subset of fields

Example Delete Statement for Postgres

- `sqlStatement := `DELETE FROM books WHERE id = $1``
- Deletes row from “table1” that db is connected to for object whose id is value represented by \$1

How to execute SQL statement from Go

- First generate string that represents the SQL statement (what the previous slides have demonstrated)
- Next execute the following command:

```
_, err := db.Exec(sqlStatement, param1, param2,...)
```

– pass as many values that your query needs
- Always return the err – can be nil
- Err will be nil if statement is executed successfully