

# Lab 09

# AWS RDS Review

WebApp

DataLab, CS, NTHU

2019 Spring

# weathermood-server

- From file to database system

```
function list(searchText = '') {
  return new Promise((resolve, reject) => {
    if (!fs.existsSync('data-posts.json')) {
      fs.writeFileSync('data-posts.json', '');
    }

    fs.readFile('data-posts.json', 'utf8', (err, data) => {
      if (err) reject(err);

      let posts = data ? JSON.parse(data) : [];
      if (posts.length > 0 && searchText) {
        posts = posts.filter(p => {
          return p.text.toLowerCase().indexOf(searchText.toLowerCase()) !== -1
        });
      }
      resolve(posts);
    });
  });
}
```

branch file

```
export function listPosts(searchText = '', start) {
  let url = `${postBaseUrl}/posts`;
  let query = [];
  if (searchText)
    query.push(`searchText=${searchText}`);
  if (start)
    query.push(`start=${start}`);
  if (query.length)
    url += '?' + query.join('&');

  console.log(`Making GET request to: ${url}`);

  return axios.get(url).then(function(res) {
    if (res.status !== 200)
      throw new Error(`Unexpected response code: ${res.status}`);
    return res.data;
  });
}
```

branch db

# lab-weathermood-server-file-todo

- Create RDS instance in AWS
  - PostgresDB is recommended in this assignment
- Set public accessibility to yes
- Configure security group

Public accessibility [Info](#)

☒ Yes  
EC2 instances and devices outside of the VPC hosting the DB instance will connect to the DB instances. You must also select one or more VPC security groups that specify which EC2 instances and devices can connect to the DB instance.

☐ No  
DB instance will not have a public IP address assigned. No EC2 instance or devices outside of the VPC will be able to connect.

PostgreSQL ▾	TCP	5432	My IP ▾	27.242.40.18/32	e.g. SSH for Admin Desktop	✕
Add Rule						

# lab-weathermood-server-file-todo

- In config.js, enter your RDS endpoint, username and password

```
switch (process.env.NODE_ENV) {  
  case 'development':  
    process.env.DB_URL =  
      `postgres://${process.env.PG_USERNAME}@${process.env.PG_HOSTNAME}:${process.env.PG_PORT}/${process.env.PG_DB_NAME}`;  
    break;  
  default: // 'staging' or 'production'  
    process.env.DB_URL =  
      `postgres://${process.env.RDS_USERNAME}:${process.env.RDS_PASSWORD}@${process.env.RDS_HOSTNAME}:${process.env.RDS_PORT}/${process.env.RDS_DB_NAME}`;  
    break;  
}
```

- You can configure them using environment variables

## 環境屬性

下列屬性會以環境屬性形式傳入應用程式。 [進一步了解](#)

名稱	值
<input type="text" value="RDS_HOSTNAME"/>	<input type="text" value="cloudprog.cxh5ndo9uvnh.us-e"/> ✕
<input type="text" value="RDS_PORT"/>	<input type="text" value="5432"/> ✕

# lab-weathermood-server-file-todo

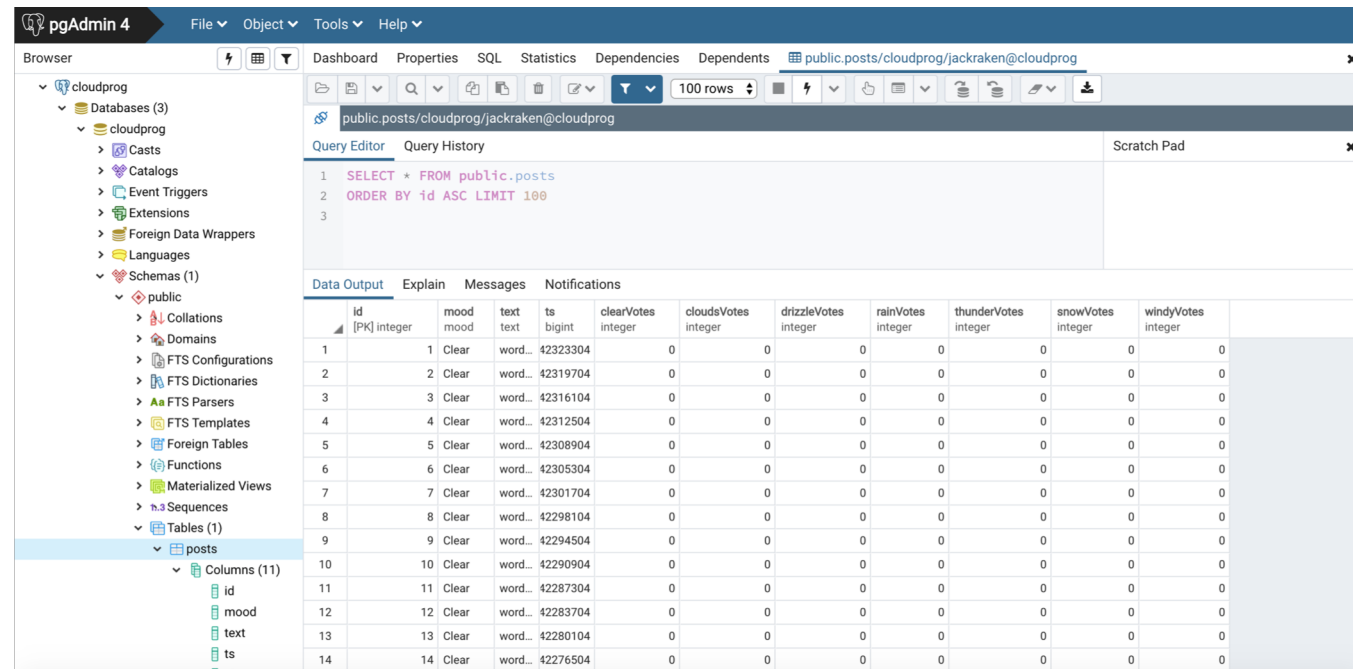
- Create the schema before running the server

```
-- Create
CREATE TYPE mood AS ENUM (
    'Clear',
    'Clouds',
    'Drizzle',
    'Rain',
    'Thunder',
    'Snow',
    'Windy'
);
CREATE TABLE posts (
    id          serial PRIMARY KEY NOT NULL,
    mood        mood NOT NULL,
    text        text NOT NULL,
    ts          bigint NOT NULL DEFAULT (extract(epoch from now())),
    "clearVotes" integer NOT NULL DEFAULT 0,
    "cloudsVotes" integer NOT NULL DEFAULT 0,
    "drizzleVotes" integer NOT NULL DEFAULT 0,
    "rainVotes" integer NOT NULL DEFAULT 0,
    "thunderVotes" integer NOT NULL DEFAULT 0,
    "snowVotes" integer NOT NULL DEFAULT 0,
    "windyVotes" integer NOT NULL DEFAULT 0
);
CREATE INDEX posts_idx_ts ON posts USING btree(ts);
CREATE INDEX posts_idx_text ON posts USING gin(text gin_trgm_ops);
```

- npm run schema

# lab-weathermood-server-file-todo

- In this assignment , You might need to create new tables or test SQL queries
- You can use pgAdmin to accomplish this



The screenshot shows the pgAdmin 4 web interface. On the left, the 'Browser' pane shows the database structure: 'cloudprog' > 'Databases (3)' > 'cloudprog' > 'Schemas (1)' > 'public' > 'Tables (1)' > 'posts'. The 'posts' table is selected, showing its columns: id, mood, text, ts, clearVotes, cloudsVotes, drizzleVotes, rainVotes, thunderVotes, snowVotes, and windyVotes. The main pane shows the 'Query Editor' with a SQL query: `SELECT * FROM public.posts ORDER BY id ASC LIMIT 100`. Below the query editor, the 'Data Output' tab displays the results of the query in a table format. The table has 11 columns: id, mood, text, ts, clearVotes, cloudsVotes, drizzleVotes, rainVotes, thunderVotes, snowVotes, and windyVotes. The results show 14 rows of data, with the first row having id 1, mood 'Clear', and text 'word...'. The table is sorted by id in ascending order.

id	mood	text	ts	clearVotes	cloudsVotes	drizzleVotes	rainVotes	thunderVotes	snowVotes	windyVotes
1	Clear	word...	42323304	0	0	0	0	0	0	0
2	Clear	word...	42319704	0	0	0	0	0	0	0
3	Clear	word...	42316104	0	0	0	0	0	0	0
4	Clear	word...	42312504	0	0	0	0	0	0	0
5	Clear	word...	42308904	0	0	0	0	0	0	0
6	Clear	word...	42305304	0	0	0	0	0	0	0
7	Clear	word...	42301704	0	0	0	0	0	0	0
8	Clear	word...	42298104	0	0	0	0	0	0	0
9	Clear	word...	42294504	0	0	0	0	0	0	0
10	Clear	word...	42290904	0	0	0	0	0	0	0
11	Clear	word...	42287304	0	0	0	0	0	0	0
12	Clear	word...	42283704	0	0	0	0	0	0	0
13	Clear	word...	42280104	0	0	0	0	0	0	0
14	Clear	word...	42276504	0	0	0	0	0	0	0

## Assignment - Weathermood Server DB-TODO

---

In this assignment, you are asked to improve your todo function using relational database.

### Requirement

---

1. Must Deploy to AWS(Must!!! or you will get 0. You may need to use AWS RDS)
2. (80%) Store and get data from DB(So you maybe need to design the DB schema).
3. (20%) Must have pagination function.

*You still need to use redux*

### Submit

---

1. One submission per team.
2. Submit *BOTH* client / server side code to GitLab.
3. Add another readme file which notes :
  1. Your team member's contribution.
  2. Your website url.

### Deadline

---

Submit your work before 2019/05/16 (Thu.) 23:59:59