강의 Flow & Plan

MQ 센서

Lecture 3

ESP32 board

Lecture 4

양방향 통신(IOT)

- MQTT
- WIFI
- LoRa
- 블루투스 Lecture 8
- RS485/RS232
- OTA Lecture 9

라즈베리파이

Lecture 2

노드레드

MQTT

데이터저장

데이터베이스

mogoDB/MySQL

데이터 분석

Lecture 5-6

- Python/Jupyter notebook
- 머신러닝/뉴럴네트워크
- 패턴인식

제어/모니터링

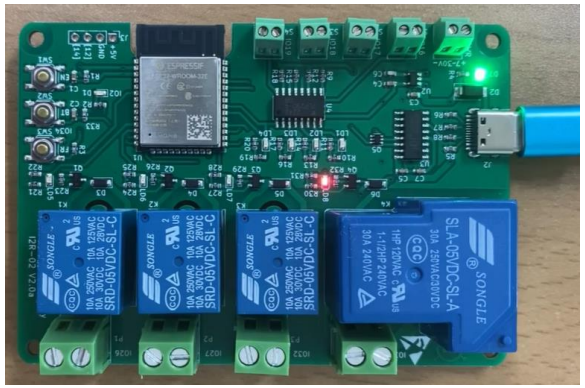
노드레드/Grafana

Lecture 7

Esp32-Cam

Lecture 14

IoT 보드 Lecture 13



UI (User Interface)

1) 웹기반

- 리눅스
- 플랫폼 : React, Next.js, Flask, Django, HTML/CSS, 노드레드

2) 모바일 기반 Lecture 10

- 안드로이드, iOS
- 플랫폼 : Flutter, React-Native, App Inventor, Blynk, Ionic

3) PC 기반 Lecture 11

- 윈도우
- 플랫폼 : C#, 닷넷

4) IOT 클라우드 (AWS) Lecture 12

- 보안적용

Lecture 15 - 16

팀프로젝터 Team1, Team2

Docker 설치

 # System update

```
sudo apt-get update
sudo apt-get upgrade
```

 # docker setup파일 다운로드

```
curl -fsSL https://get.docker.com -o get-docker.sh
```

 # Shell 실행

```
sudo sh get-docker.sh
```

 # 권한 부여

```
sudo usermod -aG docker ${USER}
```

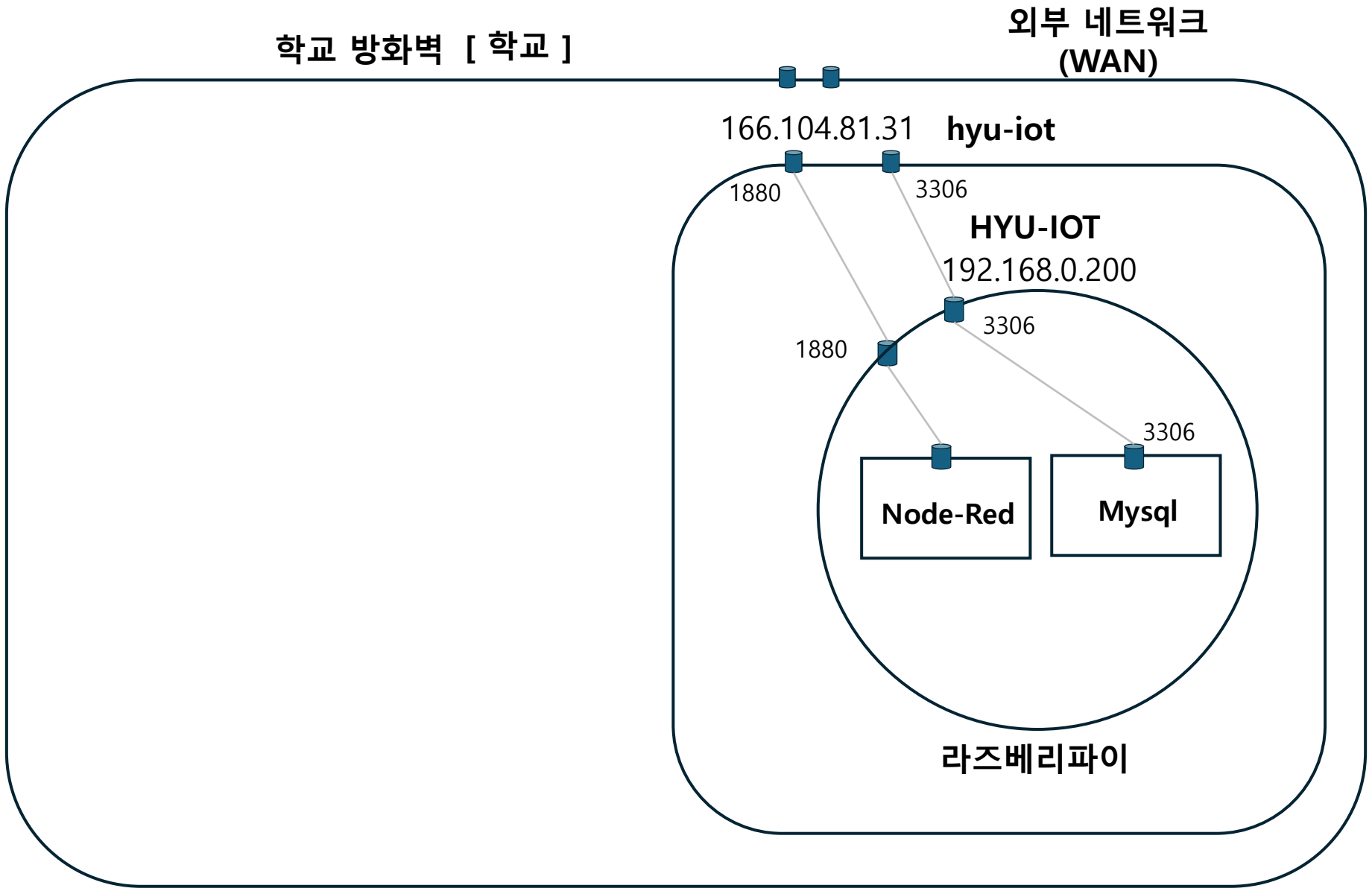
 # 권한 확인

```
groups ${user}
```

 # 부팅시 docker 자동 실행

sudo systemctl enable docker

네트워크 구조


학교 방화벽 [학교]
외부 네트워크
(WAN)
166.104.81.31
hyu-iot
1880
3306
HYU-IOT
192.168.0.200
3306
1880
3306
Node-Red
Mysql
라즈베리파이

Container Application in Docker

https://www.docker.com/products/docker-hub/

Node-Red 서버

$ docker run -itd -p 1880:1880 -v /home/iot/nodered_1880:/data --name nodered_1880 nodered/node-red:latest

Mysql 서버

https://dev.mysql.com/downloads/installer/

$ docker run -itd -p 3306:3306 --name mysql_3306 -e MYSQL_ROOT_PASSWORD=iot1004 mysql:latest

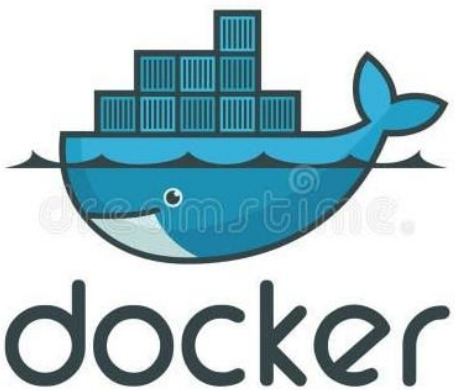
monoDB 서버

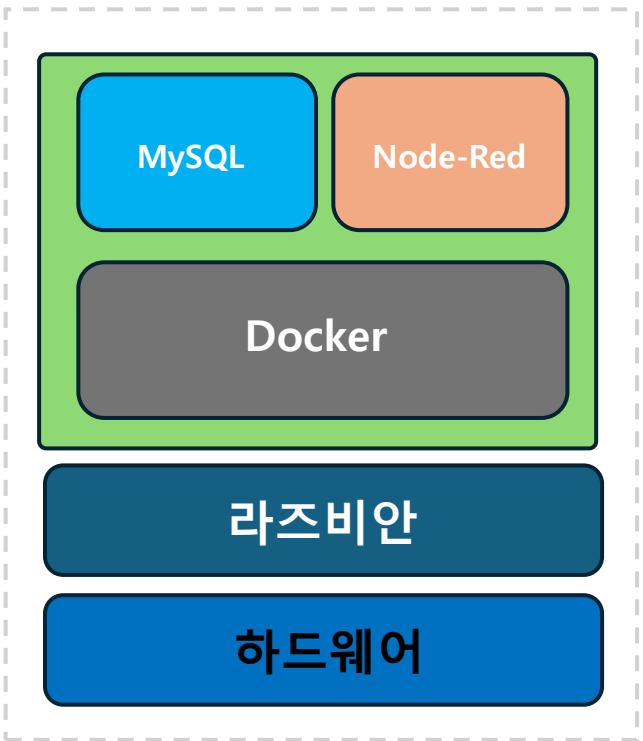
$ docker run -itd -p 27017:27017 --name mongodb mongodb/mongodb-community-server:latest

MQTT 서버

$ docker run -it -p 1883:1883 -p 9001:9001 -v mosquitto.conf:/mosquitto/config/mosquitto.conf eclipse-mosquitto

```
iot@HYU-IOT: ~
iot@HYU-IOT:~ $ docker ps -a
CONTAINER ID   IMAGE              COMMAND                  CREATED        STATUS                  PORTS                                                  NAMES
0734fcb762e0   mysql:latest       "docker-entrypoint.s…"   10 hours ago   Up 7 hours              0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp   mysql_3306
7f0c0f92c3f4   nodered/node-red   "./entrypoint.sh"        10 hours ago   Up 7 hours (healthy)    0.0.0.0:1880->1880/tcp, :::1880->1880/tcp              nodered_1880
iot@HYU-IOT:~ $
```





MQ 센서 **ESP32 board**

Lecture 3

데이터 입력/출력

양방향 통신(IOT)

MQTT

라즈베리파이

제어 컨트롤러

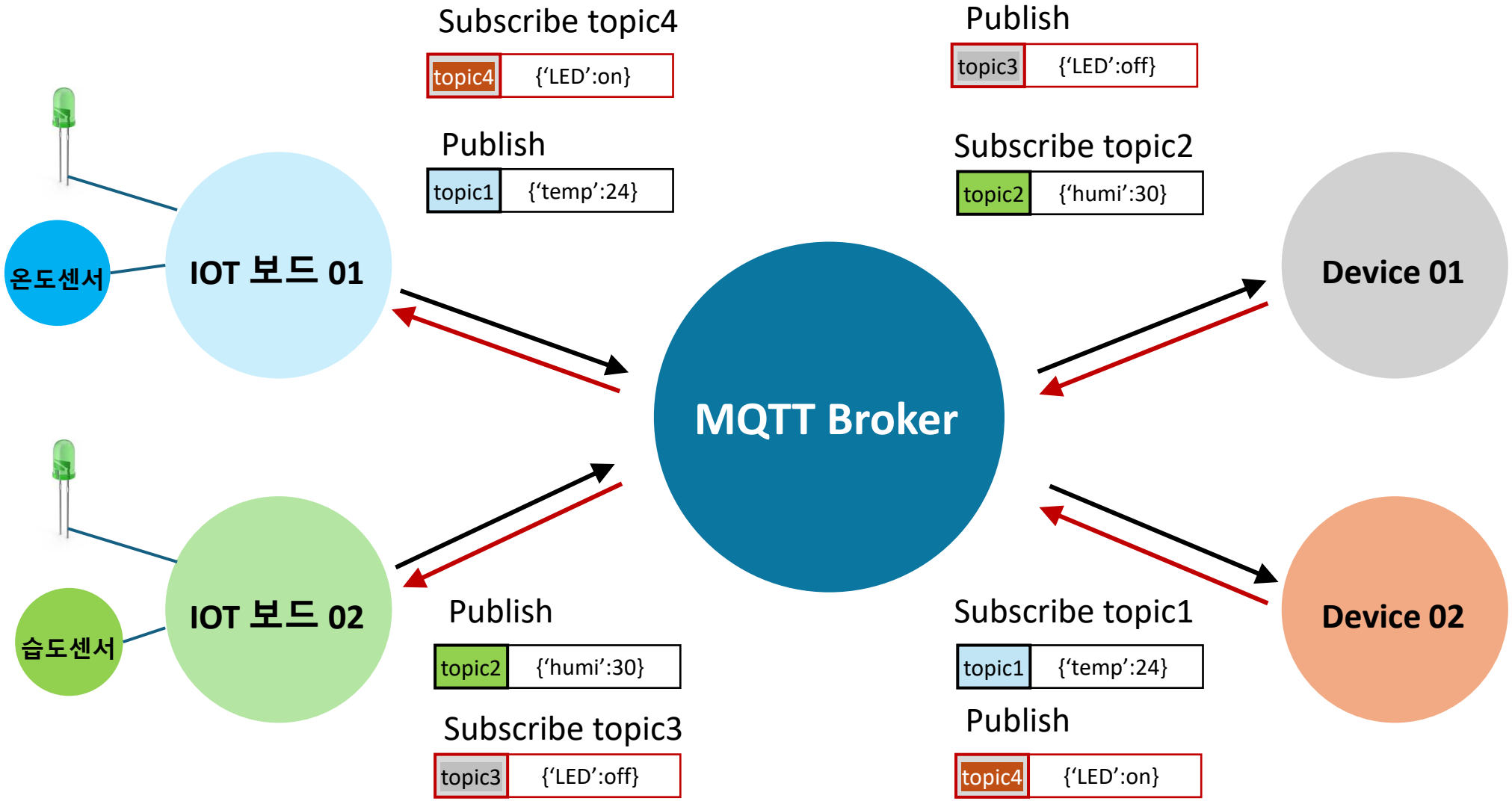
Node-RED

데이터저장

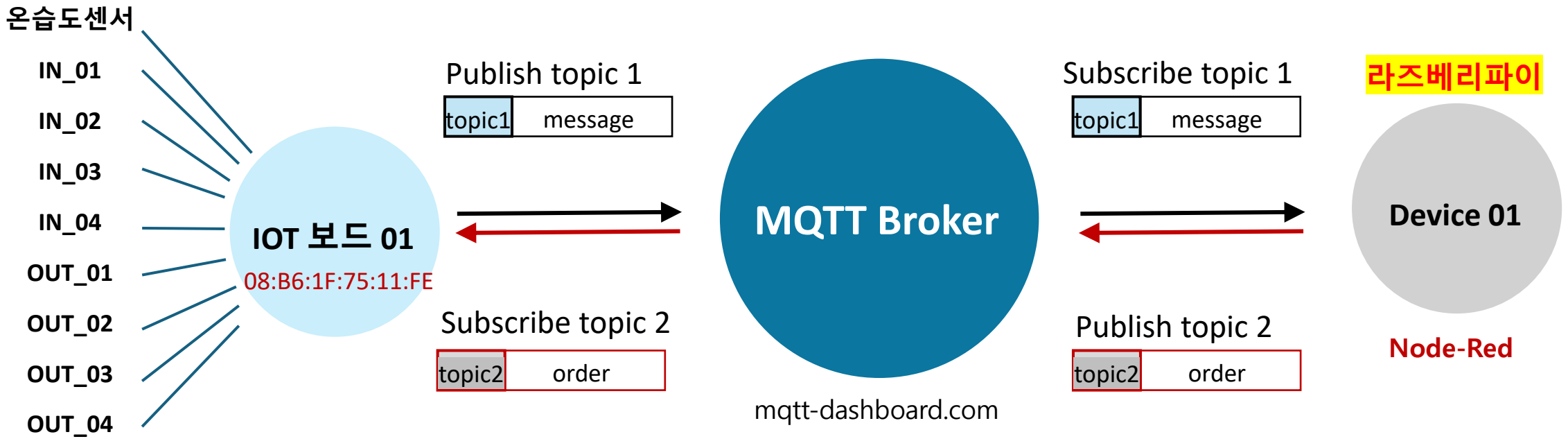
데이터저장

MySQL

MQTT 통신


Subscribe topic4
topic4
{'LED':on}
Publish
topic1
{'temp':24}
Publish
topic3
{'LED':off}
Subscribe topic2
topic2
{'humi':30}
온도센서
IOT 보드 01
MQTT Broker
Device 01
습도센서
IOT 보드 02
Device 02
Publish
topic2
{'humi':30}
Subscribe topic3
topic3
{'LED':off}
Subscribe topic1
topic1
{'temp':24}
Publish
topic4
{'LED':on}

IOT 보드 MQTT 통신 모식도



Data

topic1	message

Order

topic2	message

Topic	Message
i2r/bombzoo78@gmail.com/out	{"type":3,"email":"bombzoo78@gmail.com","mac":"EC:64:C9:43:E8:B8", "temp":28.4,"humi":38,"in":[0,0,0,0],"out":[0,0,0,0]}
i2r/bombzoo78@gmail.com/in	{"mac":"08:B6:1F:75:11:FE","order":2,"no":1,"value":true}

IOT 보드

Broker

- mqtt-dashboard.com

Topic

- Out : i2r/bombzoo78@gmail.com/out
- In : i2r/bombzoo78@gmail.com/in

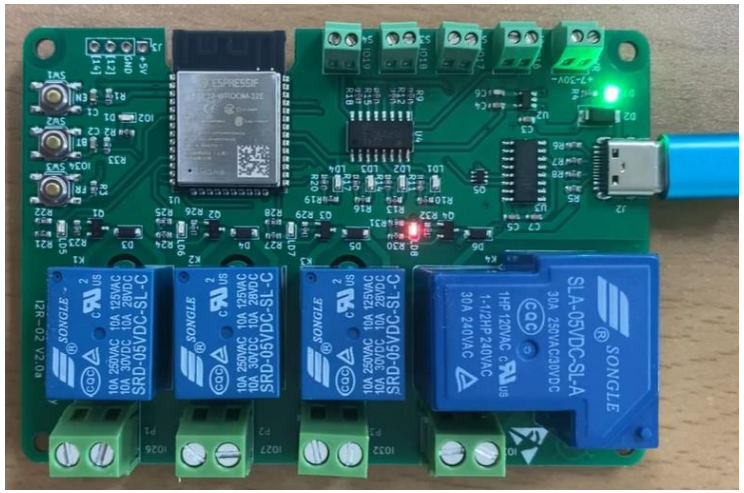
Mac Address

"08:B6:1F:75:11:FE"

order	기능	설명 및 프로토콜
0	펌웨어 다운로드	인터넷에서 통신으로 펌웨어를 보드로 내려 받는다 **{'order':0, 'fileName'='i2r-03.ino.bin'}**
1	정보입력	와이파이 및 통신에 필요한 정보를 보내 보드에 기록하여 기기는 여기 정보로 통신을 연결한다. **{"order":1, 'ssid':'',*'password'=*", 'email':'bombzoo78@gmail.com', 'mqttBroker':' mqtt-dashboard.com'}**
2	핀출력	IoT PLC의 핀번호(0,1,2,3 4개)와 true/false를 보내면 릴레이가 동작한다. **{"mac":"08:B6:1F:75:11:FE","order":2,"no":1,"value":true}** 맥어드레스가 "08:B6:1F:75:11:FE"인 기기의 1번핀 릴레이를 on 시킨다.
3	상태전송요청	요청 : **{"mac":"EC:64:C9:43:E8:B8",'order':3}** 응답예시 **{"type":3,"email":"bombzoo78@gmail.com","mac":"EC:64:C9:43:E8:B8","temp":28.4,"humi":38,"in":[0,0,0,0],"out":[0,0,0,0]}**

네트워크 구조

MQTT
Broker



[학교]

166.104.81.31 hyu-iot

1880 3306

HYU-IOT

192.168.0.200

1880 3306

1880 3306

Node-Red Mysql

라즈베리파이

라즈베리파이

데이터 입력/출력

양방향 통신(IOT)

MQTT

제어 컨트롤러

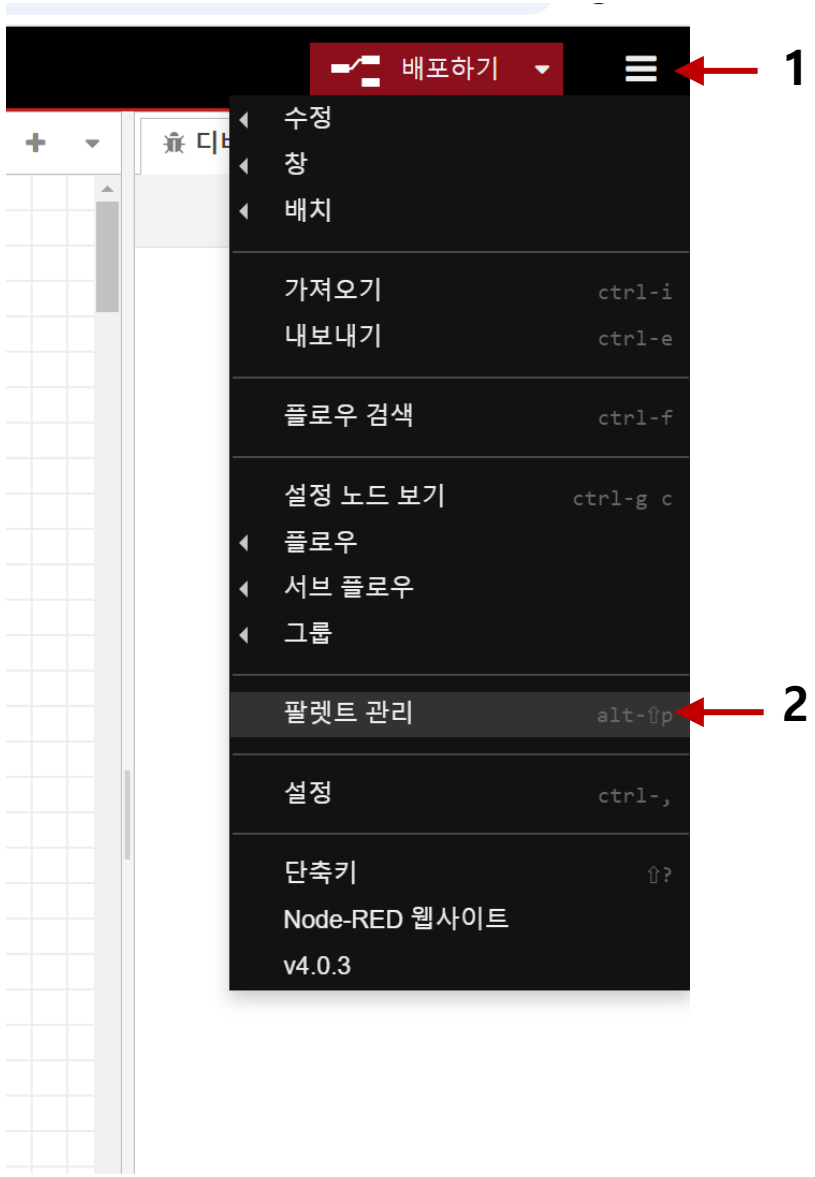
Node-RED

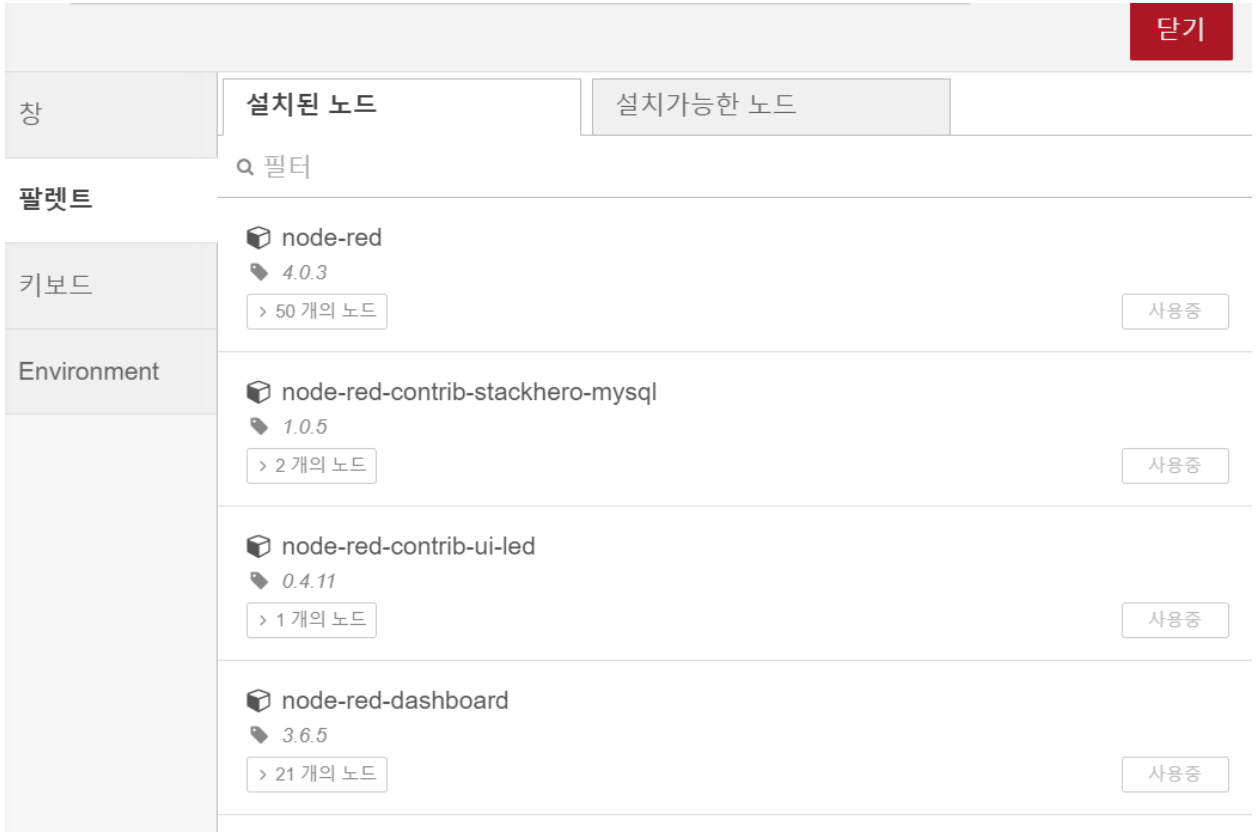
데이터저장

데이터저장

MySQL

Node-Red 팔레트 설치

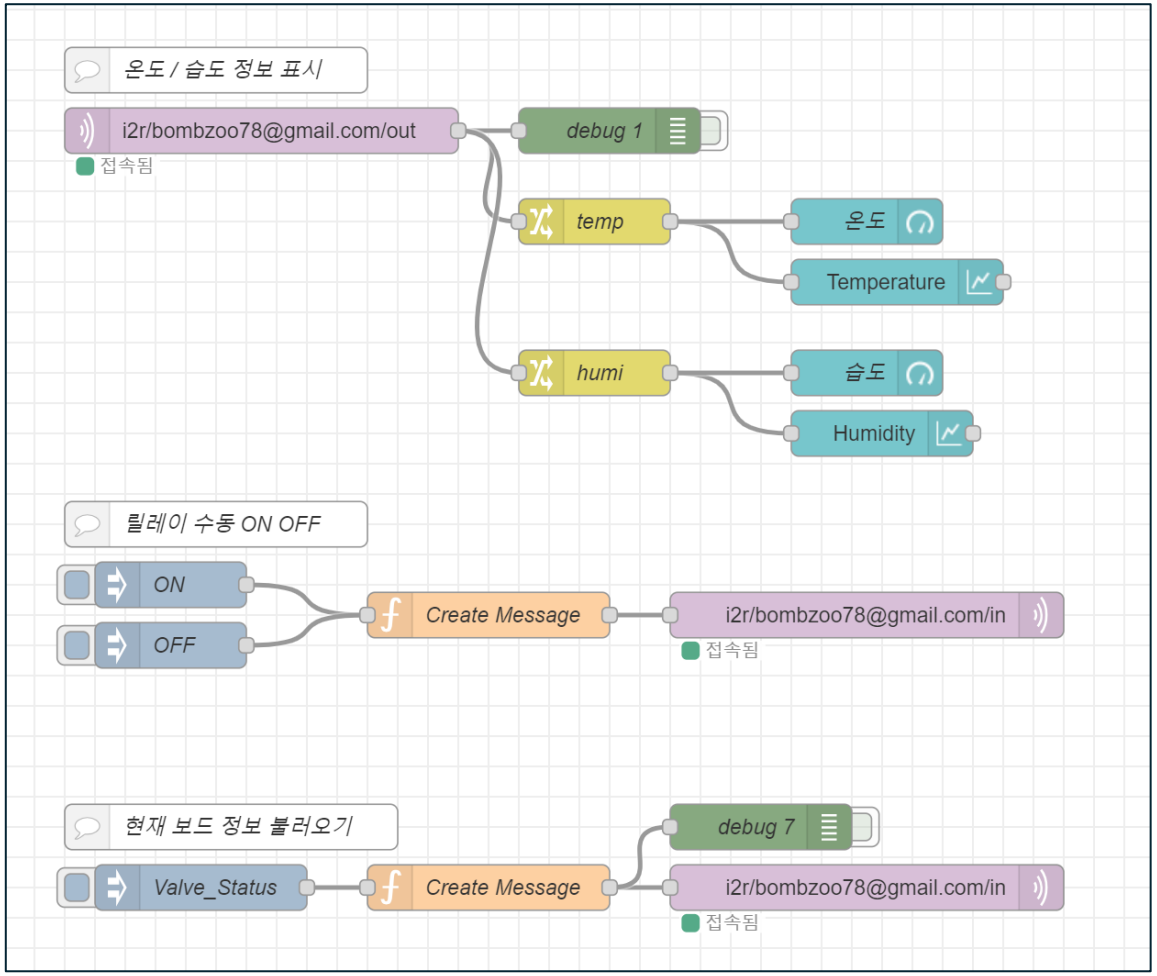




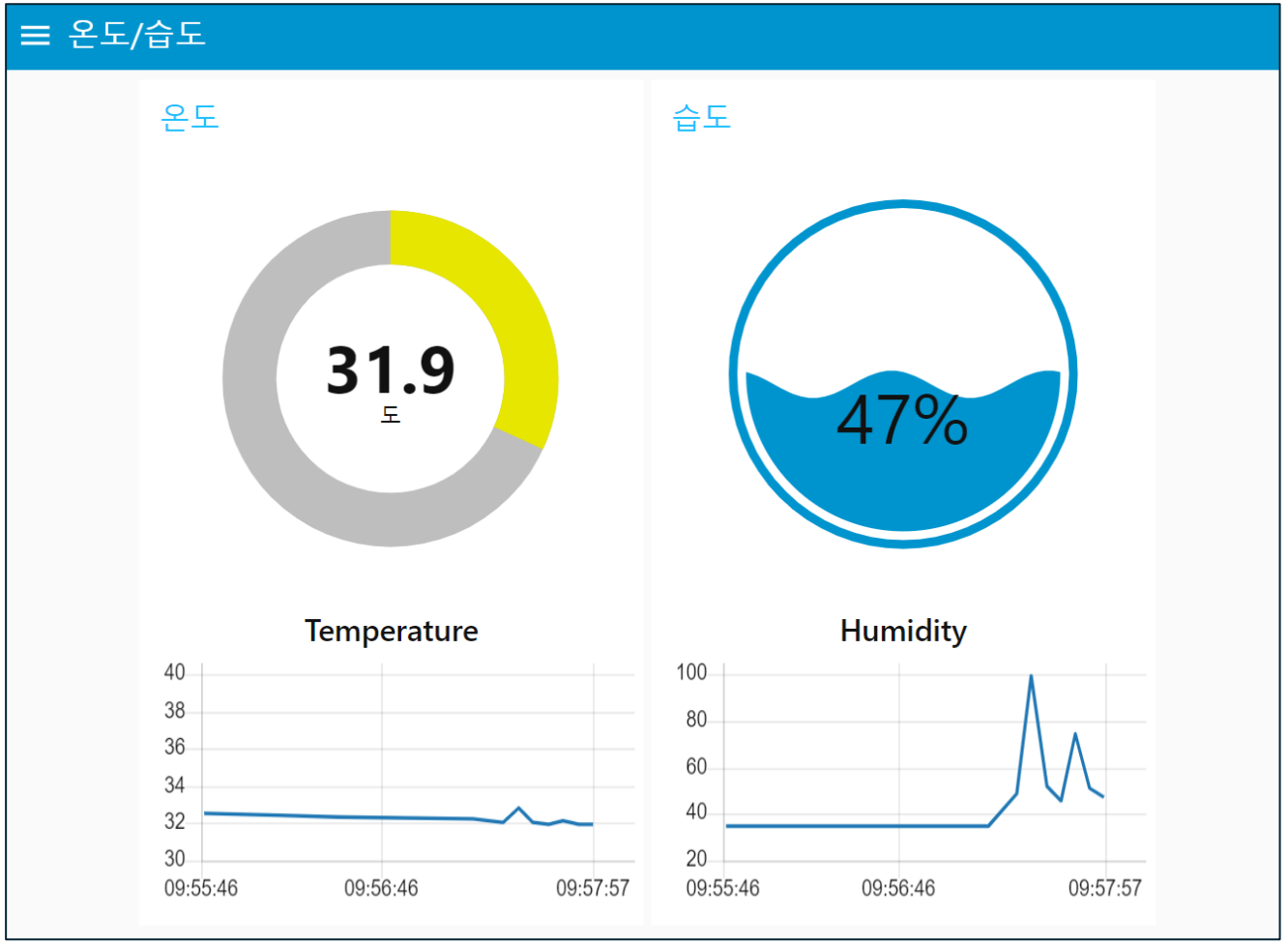
1. **node-red-dashboard**
2. **node-red-contrib-ui-led**
3. **node-red-contrib-stackhero-mysql**

Node-Red 사용법

온도/습도 표시

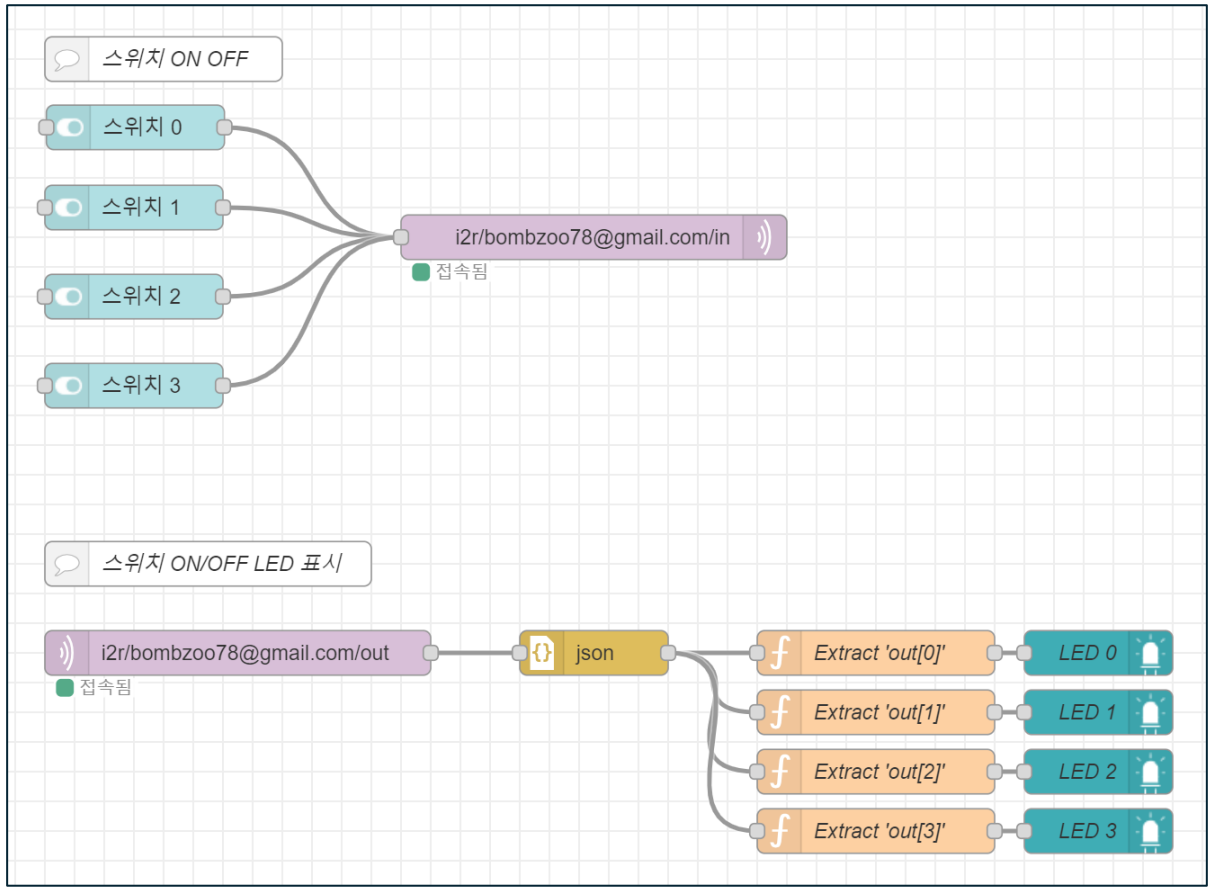


Dashboard

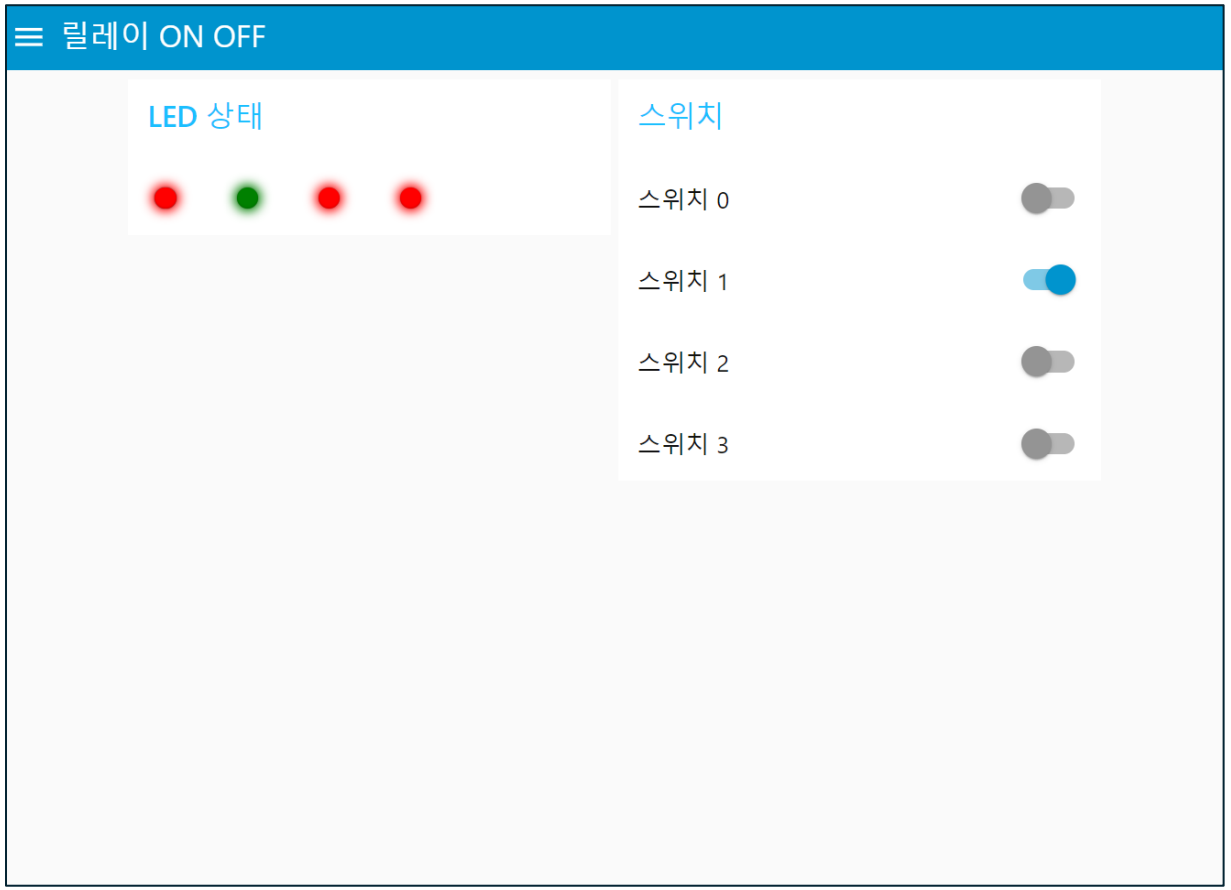


Node-Red 사용법

출력 스위치 on/off

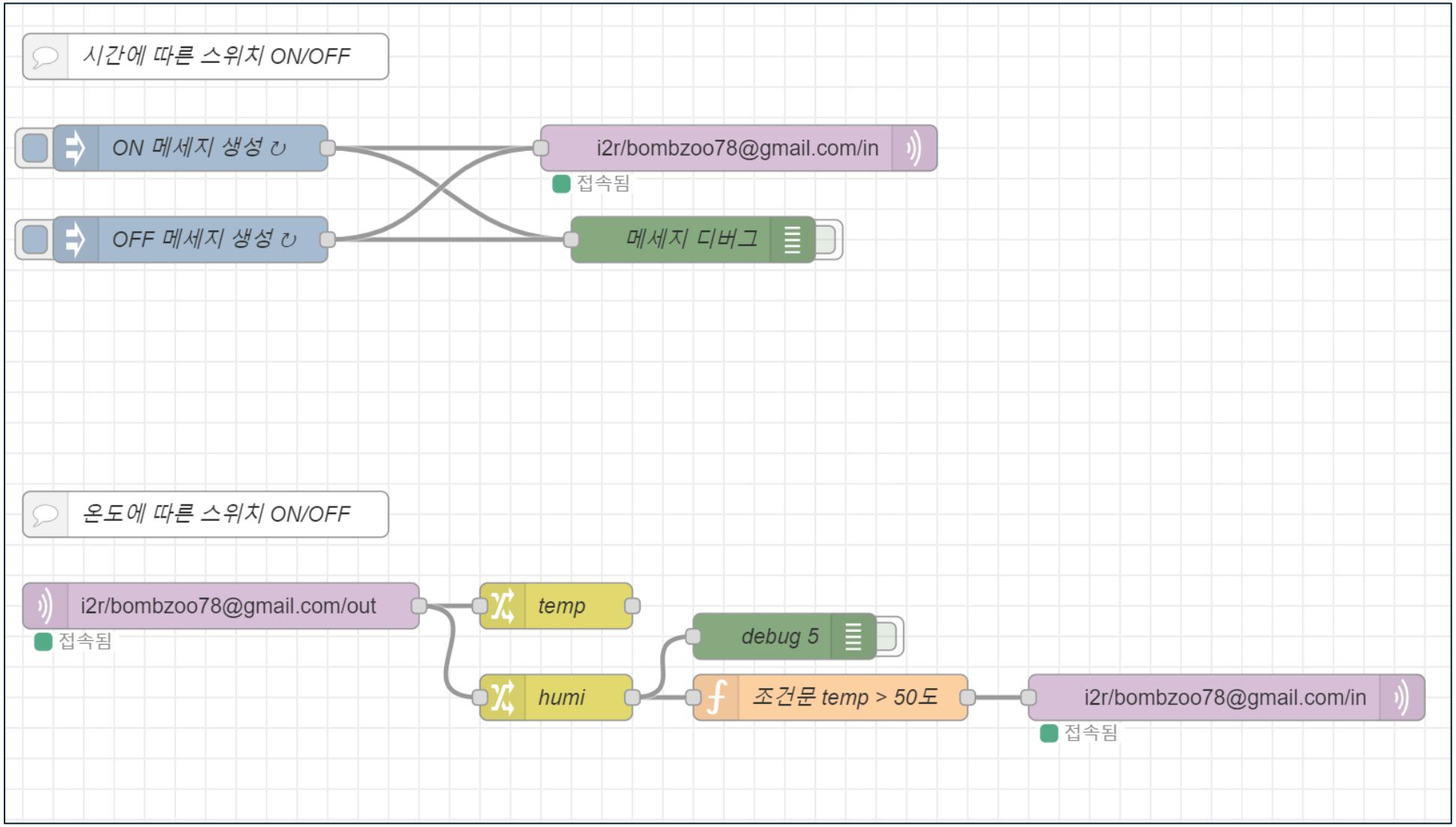


Dashboard

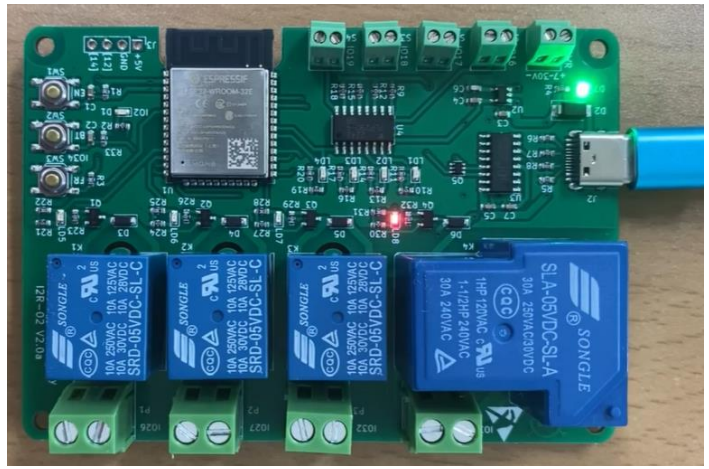


Node-Red 사용법

스케줄러



데이터 입력/출력



양방향 통신(IOT)

MQTT

라즈베리파이

제어 컨트롤러

Node-RED

데이터저장

데이터저장

MySQL

MySQL 컨테이너 접속

라즈비안

컨테이너

mysql

```
iot@HYU-IOT:~ $ docker ps -a
CONTAINER ID   IMAGE                COMMAND                  CREATED         STATUS
   NAMES
0734fcb762e0   mysql:latest         "docker-entrypoint.s…"   10 hours ago    Up 7 hours
   mysql_3306
7f0c0f92c3f4   nodered/node-red     "./entrypoint.sh"        10 hours ago    Up 7 hours (healt
   nodered_1880
iot@HYU-IOT:~ $ docker exec -it mysql_3306 /bin/bash
bash-5.1# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 151
Server version: 9.0.1 MySQL Community Server - GPL

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

SQL 분류

데이터 정의어	데이터 조작어	데이터 제어어
테이블을 생성하고 변경, 삭제하는 기능을 제공	테이블에 새 데이터를 삽입하거나, 저장된 데이터를 수정, 삭제, 검색하는 기능을 제공	보안을 위해 데이터에 대한 접근 및 사용 권한을 사용자별로 부여하거나 취소하는 기능을 제공
• CREATE • ALTER • DROP	• INSERT • UPDATE • DELETE • SELECT	• GRANT • REVOKE

1. 데이터 정의 언어

 # 데이터베이스 리스트 보기

```
SHOW DATABASES;
```

 # 데이터베이스 만들기

```
CREATE DATABASE my_database;
```

 # 데이터베이스 지우기

```
DROP DATABASE my_database;
```

 # 데이터베이스 사용하기

```
USE my_database;
```

 # 테이블 만들기

```
CREATE TABLE users (
  id INT PRIMARY KEY,
  name VARCHAR(100),
  age INT
);
```

```
mysql> select * from users;
+----+-------+------+-------+
| id | name  | age  | email |
+----+-------+------+-------+
|  1 | Alice |   30 | NULL  |
+----+-------+------+-------+
1 row in set (0.00 sec)
```

 # 테이블 수정하기

```
ALTER TABLE users ADD COLUMN email VARCHAR(100);
```

 # 테이블 지우기

```
DROP TABLE users;
```

데이터 타입	설명	예시	크기/제한사항
INT	정수 값을 저장 (음수, 양수 모두 가능)	age INT	-2147483648 ~ 2147483647
TINYINT	작은 범위의 정수 값을 저장 (1바이트)	status TINYINT	-128 ~ 127 또는 0 ~ 255
SMALLINT	작은 범위의 정수 값을 저장 (2바이트)	year SMALLINT	-32768 ~ 32767
BIGINT	큰 범위의 정수 값을 저장 (8바이트)	population BIGINT	-2^63 ~ 2^63-1
FLOAT	부동소수점 숫자를 저장 (소수점 이하 포함)	price FLOAT	4바이트, 소수점 이하 7자리 정밀도
DOUBLE	큰 범위의 부동소수점 숫자를 저장 (더 높은 정밀도)	temperature DOUBLE	8바이트, 소수점 이하 15자리 정밀도
DECIMAL	고정된 소수점 숫자를 저장 (정확한 소수점 이하 자리수 지정 가능)	salary DECIMAL(10,2)	총 10자리, 소수점 이하 2자리
CHAR(n)	고정 길이의 문자열 저장 (정확히 n 자리의 문자)	code CHAR(5)	최대 255자
VARCHAR(n)	가변 길이의 문자열 저장 (최대 n 자리까지)	name VARCHAR(100)	최대 65535자
TEXT	매우 큰 텍스트 데이터를 저장	description TEXT	최대 65535자
MEDIUMTEXT	큰 텍스트 데이터를 저장	bio MEDIUMTEXT	최대 16,777,215자
LONGTEXT	매우 큰 텍스트 데이터를 저장	article LONGTEXT	최대 4,294,967,295자
DATE	날짜를 저장 (형식: YYYY-MM-DD)	birthdate DATE	1000-01-01 ~ 9999-12-31
TIME	시간을 저장 (형식: HH:MM:SS)	appointment TIME	-838:59:59 ~ 838:59:59
DATETIME	날짜와 시간을 저장 (형식: YYYY-MM-DD HH:MM:SS)	created_at DATETIME	1000-01-01 00:00:00 ~ 9999-12-31 23:59:59
TIMESTAMP	UNIX 타임스탬프 형식으로 날짜와 시간을 저장 (기본값: 현재 시간)	updated_at TIMESTAMP	1970-01-01 00:00:01 ~ 2038-01-19 03:14:07
BOOLEAN	참(TRUE) 또는 거짓(FALSE) 값 저장	is_active BOOLEAN	0 (거짓), 1 (참)
BLOB	바이너리 데이터를 저장 (이미지, 동영상 등)	image BLOB	최대 65,535바이트
MEDIUMBLOB	큰 바이너리 데이터를 저장	video MEDIUMBLOB	최대 16MB
LONGBLOB	매우 큰 바이너리 데이터를 저장	large_file LONGBLOB	최대 4GB

2. 데이터 조작 언어

데이터 검색	SELECT
데이터 삽입	INSERT
데이터 수정	UPDATE
데이터 삭제	DELETE

 # 테이블에 데이터 첨가하기

```
INSERT INTO users (id, name, age) VALUES (1, 'Alice', 30);

INSERT INTO users (id, name, email, age) VALUES (2, 'John',
'john@example.com', 32), (3, 'Bob', 'bob@example.com', 25),
(4, 'Charlie', 'charlie@example.com', 35);
```

 # 데이터 조회 (SELECT)

```
SELECT name, age FROM users;

SELECT * FROM users;
```

 # 데이터 수정하기

```
UPDATE users SET age = 31 WHERE id = 1;

UPDATE users SET age = 26 WHERE name = 'Bob';
```

 # 데이터 삭제하기

```
DELETE FROM users WHERE name = 'Charlie';
```

3. 데이터 조회 언어

데이터 조건	WHERE
데이터 검색	LIKE
데이터 정렬	ODER BY
데이터 제한	LIMIT

 # 조건 데이터 조회 (WHERE)

```
SELECT name FROM users WHERE age > 25;
```

 # 조건 데이터 조회 (LIKE)

```
SELECT * FROM users WHERE email LIKE 'john%';
SELECT * FROM users WHERE name LIKE '%ar%';
SELECT * FROM users WHERE name LIKE '__o';
```

 # 조건 결과 정렬 (ORDER BY)

```
SELECT name, age FROM users ORDER BY age DESC;
```

 # 조건 결과 제한 (LIMIT)

```
SELECT name, age FROM users ORDER BY age DESC LIMIT 2;
```

[WHERE 사용법]

사용 예시	설명
`WHERE age = 30`	`age`가 30인 데이터를 검색
`WHERE name LIKE 'J%'`	`name`이 'J'로 시작하는 데이터를 검색
`WHERE age > 30`	`age`가 30보다 큰 데이터를 검색
`WHERE age BETWEEN 20 AND 30`	`age`가 20과 30 사이에 있는 데이터를 검색 (포함)
`WHERE name IN ('John', 'Alice', 'Bob')`	`name`이 'John', 'Alice', 'Bob' 중 하나인 데이터를 검색
`WHERE age IS NULL`	`age`가 NULL인 데이터를 검색
`WHERE city = 'New York' AND age > 25`	`city`가 'New York'이면서 `age`가 25보다 큰 데이터 검색
`WHERE city = 'New York' OR city = 'Los Angeles'`	`city`가 'New York'이거나 'Los Angeles'인 데이터 검색

[LIKE 사용법]

와일드카드 기호 설명

기호	설명
`%`	0개 이상의 문자 (문자의 내용과 개수는 상관 없음)
`_`	1개의 문자 (문자의 내용은 상관 없음)

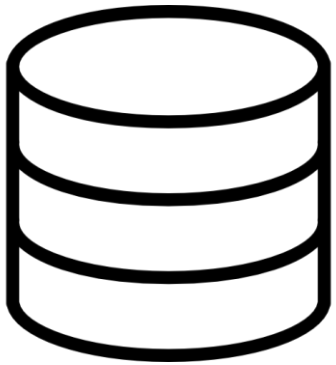
LIKE 사용 예시

사용 예시	설명
`LIKE 'data%'`	`data`로 시작하는 문자열 (길이 상관없이 `data`로 시작)
`LIKE '%data'`	`data`로 끝나는 문자열 (길이 상관없이 `data`로 끝남)
`LIKE '%data%'`	`data`가 포함된 문자열 (길이 상관없이 `data`가 포함)
`LIKE 'data____'`	`data`로 시작하는 8자리 문자열
`LIKE '___data'`	`data`로 끝나는 8자리 문자열

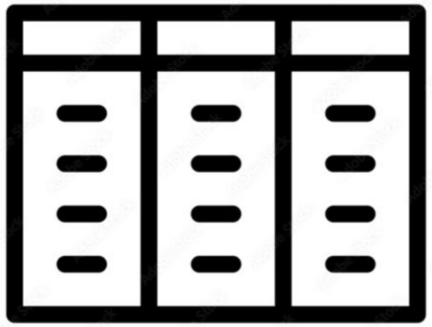
IOT DB 만들기

[IOT DB 만들기]

iot DB



status TABLE



 # 데이터베이스 만들기

CREATE DATABASE iot;

 # 데이터베이스 사용하기

USE iot;

 # 테이블 보기

SHOW TABLES;

 # 테이블 만들기

```
CREATE TABLE status (
    timestamp VARCHAR(45) PRIMARY KEY,
    email VARCHAR(45) NOT NULL,
    mac VARCHAR(45) NOT NULL,
    temp FLOAT NOT NULL,
    humi FLOAT NOT NULL,
    in00 INT NOT NULL,
    in01 INT NOT NULL,
    in02 INT NOT NULL,
    in03 INT NOT NULL,
    out00 INT NOT NULL,
    out01 INT NOT NULL,
    out02 INT NOT NULL,
    out03 INT NOT NULL
);
```

IOT DB 만들기

IOT 보드 실시간 정보

```
"type":3,
"email": "bombzoo78@gmail.com",
"mac":"EC:64:C9:43:E8:B8",
"temp":28.4,
"humi":38,
"in":[0,0,0,0],
"out":[0,0,0,0]
```

```
"timestamp " :1727170478,
"email": "bombzoo78@gmail.com",
"mac":"EC:64:C9:43:E8:B8",
"temp":28.4,
"humi":38,
"in":[0,0,0,0],
"out":[0,0,0,0]
```

 # 데이터 보기

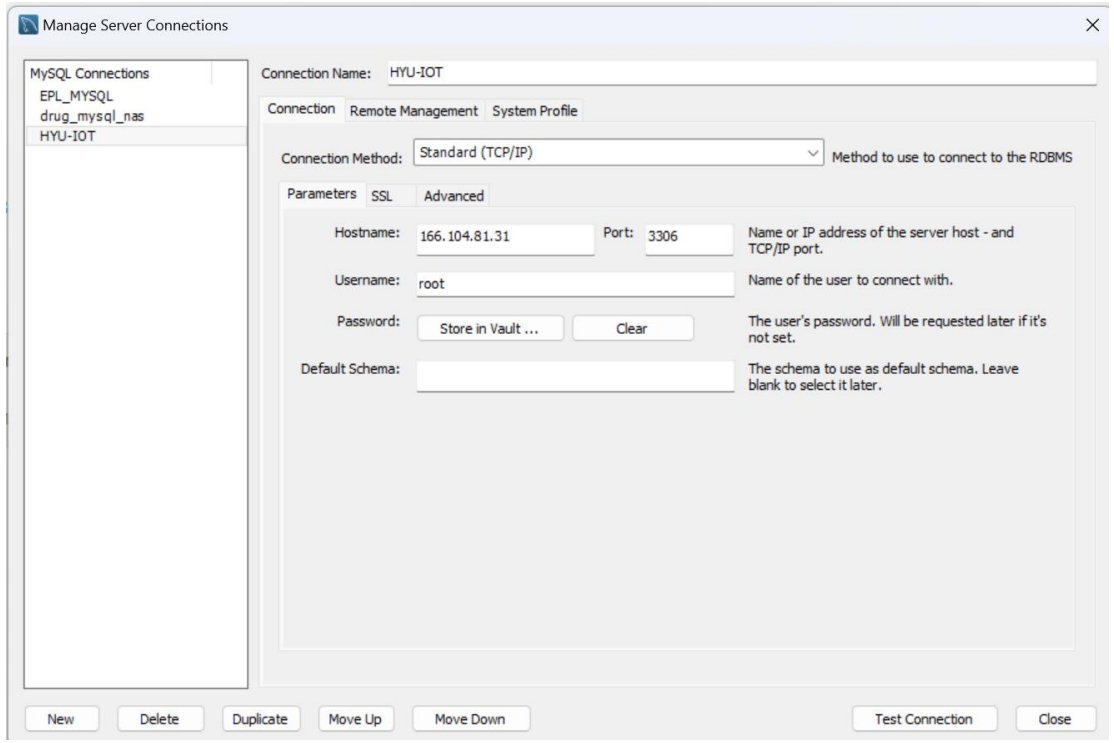
SELECT * FROM status;

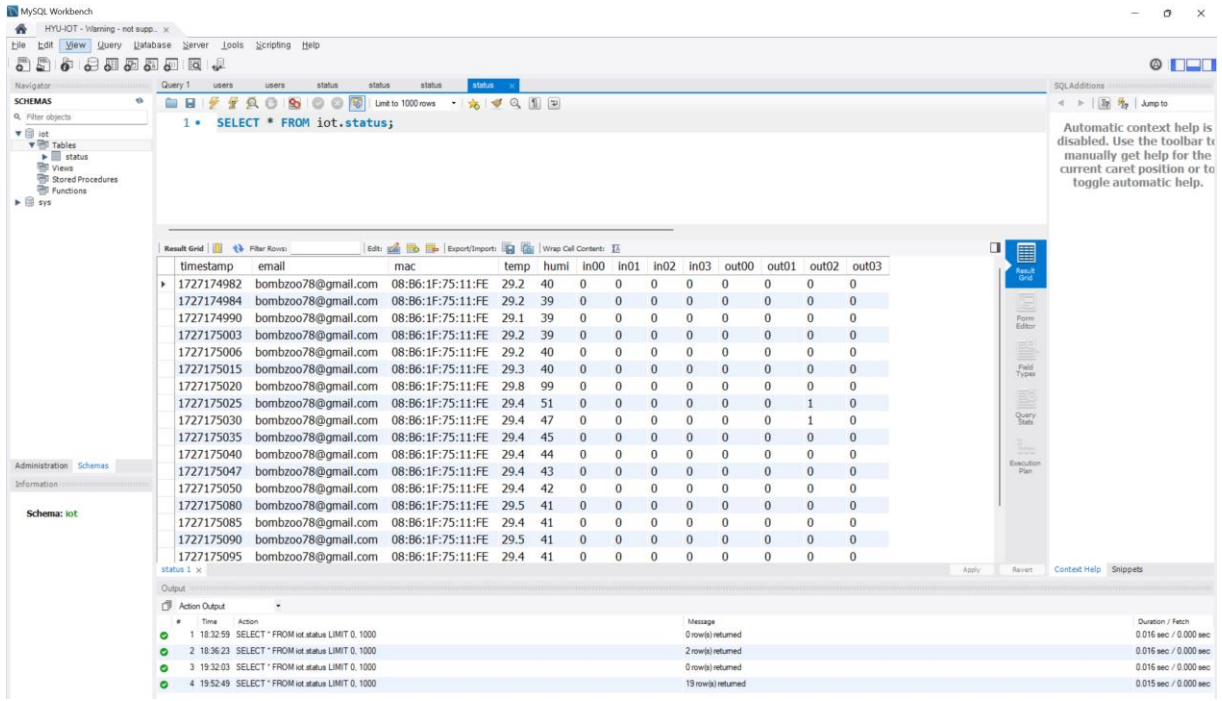
```
mysql> select * from status;
+------------+---------------------+-------------------+------+------+------+------+------+------+-------+-------+-------+-------+
| timestamp  | email               | mac               | temp | humi | in00 | in01 | in02 | in03 | out00 | out01 | out02 | out03 |
+------------+---------------------+-------------------+------+------+------+------+------+------+-------+-------+-------+-------+
| 1727174982 | bombzoo78@gmail.com | 08:B6:1F:75:11:FE | 29.2 |   40 |    0 |    0 |    0 |    0 |     0 |     0 |     0 |     0 |
| 1727174984 | bombzoo78@gmail.com | 08:B6:1F:75:11:FE | 29.2 |   39 |    0 |    0 |    0 |    0 |     0 |     0 |     0 |     0 |
| 1727174990 | bombzoo78@gmail.com | 08:B6:1F:75:11:FE | 29.1 |   39 |    0 |    0 |    0 |    0 |     0 |     0 |     0 |     0 |
| 1727175003 | bombzoo78@gmail.com | 08:B6:1F:75:11:FE | 29.2 |   39 |    0 |    0 |    0 |    0 |     0 |     0 |     0 |     0 |
| 1727175006 | bombzoo78@gmail.com | 08:B6:1F:75:11:FE | 29.2 |   40 |    0 |    0 |    0 |    0 |     0 |     0 |     0 |     0 |
| 1727175015 | bombzoo78@gmail.com | 08:B6:1F:75:11:FE | 29.3 |   40 |    0 |    0 |    0 |    0 |     0 |     0 |     0 |     0 |
| 1727175020 | bombzoo78@gmail.com | 08:B6:1F:75:11:FE | 29.8 |   99 |    0 |    0 |    0 |    0 |     0 |     0 |     0 |     0 |
| 1727175025 | bombzoo78@gmail.com | 08:B6:1F:75:11:FE | 29.4 |   51 |    0 |    0 |    0 |    0 |     0 |     0 |     1 |     0 |
| 1727175030 | bombzoo78@gmail.com | 08:B6:1F:75:11:FE | 29.4 |   47 |    0 |    0 |    0 |    0 |     0 |     0 |     1 |     0 |
| 1727175035 | bombzoo78@gmail.com | 08:B6:1F:75:11:FE | 29.4 |   45 |    0 |    0 |    0 |    0 |     0 |     0 |     0 |     0 |
| 1727175040 | bombzoo78@gmail.com | 08:B6:1F:75:11:FE | 29.4 |   44 |    0 |    0 |    0 |    0 |     0 |     0 |     0 |     0 |
| 1727175047 | bombzoo78@gmail.com | 08:B6:1F:75:11:FE | 29.4 |   43 |    0 |    0 |    0 |    0 |     0 |     0 |     0 |     0 |
| 1727175050 | bombzoo78@gmail.com | 08:B6:1F:75:11:FE | 29.4 |   42 |    0 |    0 |    0 |    0 |     0 |     0 |     0 |     0 |
```

MySQL Workbench

 # 홈페이지

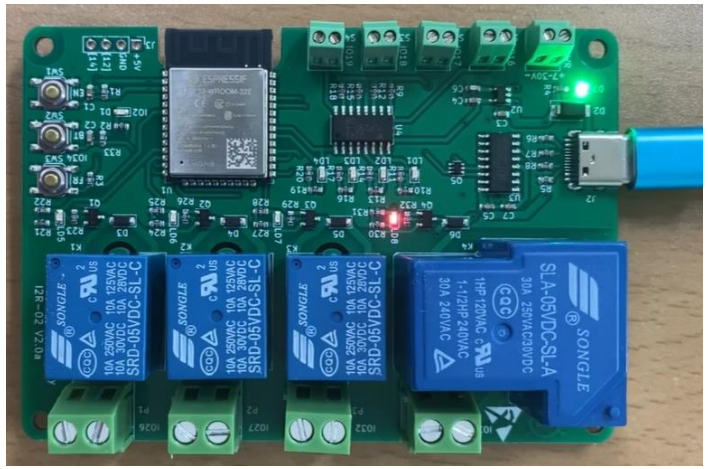
https://dev.mysql.com/downloads/workbench/





노드레드 – MySQL 연결

데이터 입력/출력



양방향 통신(IOT)

MQTT

라즈베리파이

제어 컨트롤러

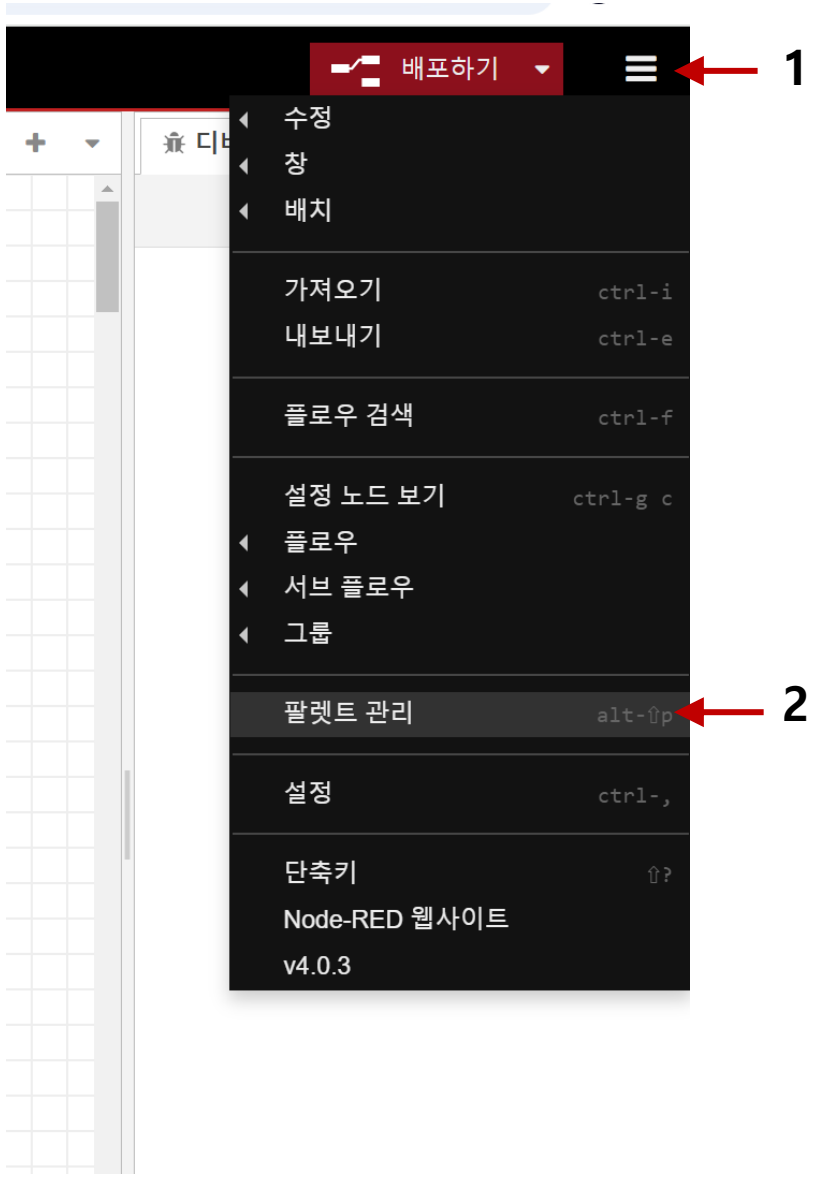
Node-RED

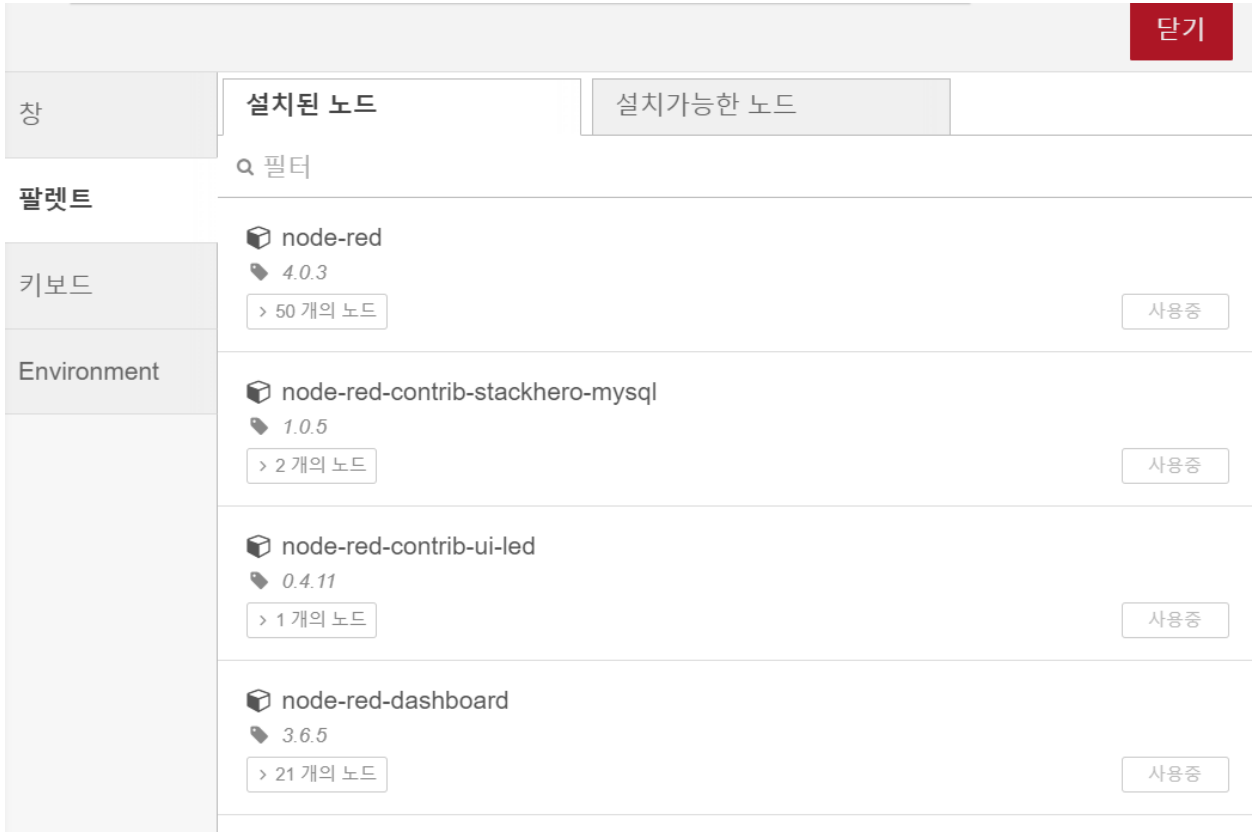
데이터저장

데이터저장

MySQL

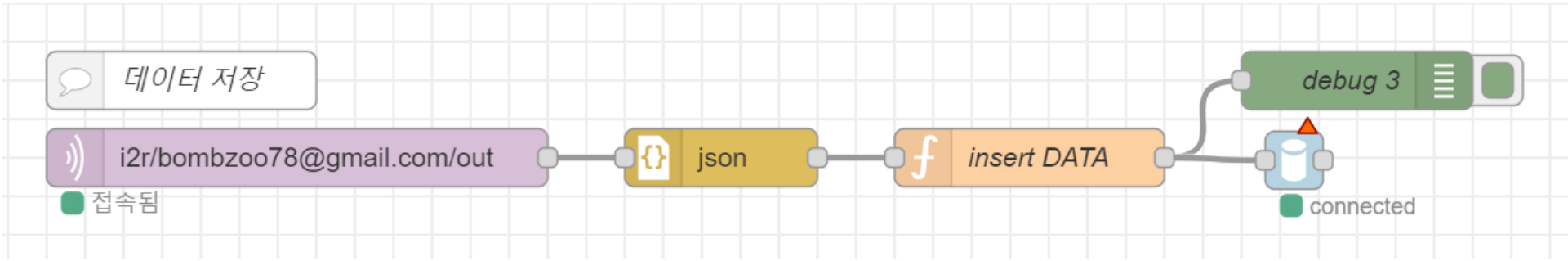
Node-Red 팔레트 설치





1. **node-red-dashboard**
2. **node-red-contrib-ui-led**
3. **node-red-contrib-stackhero-mysql**

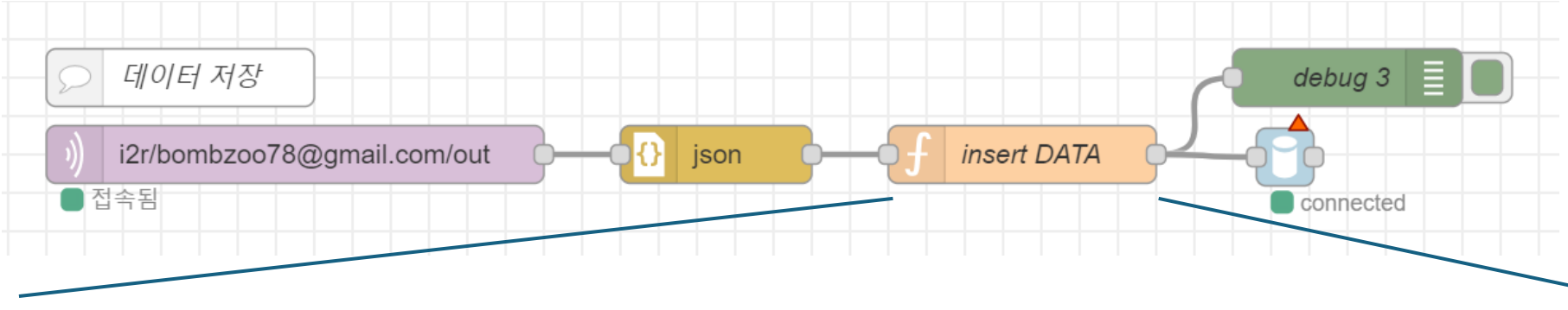
노드레드 – MySQL 연결



IOT 보드 실시간 정보

```
"type":3,
"email": "bombzoo78@gmail.com",
"mac":"EC:64:C9:43:E8:B8",
"temp":28.4,
"humi":38,
"in":[0,0,0,0],
"out":[0,0,0,0]
```

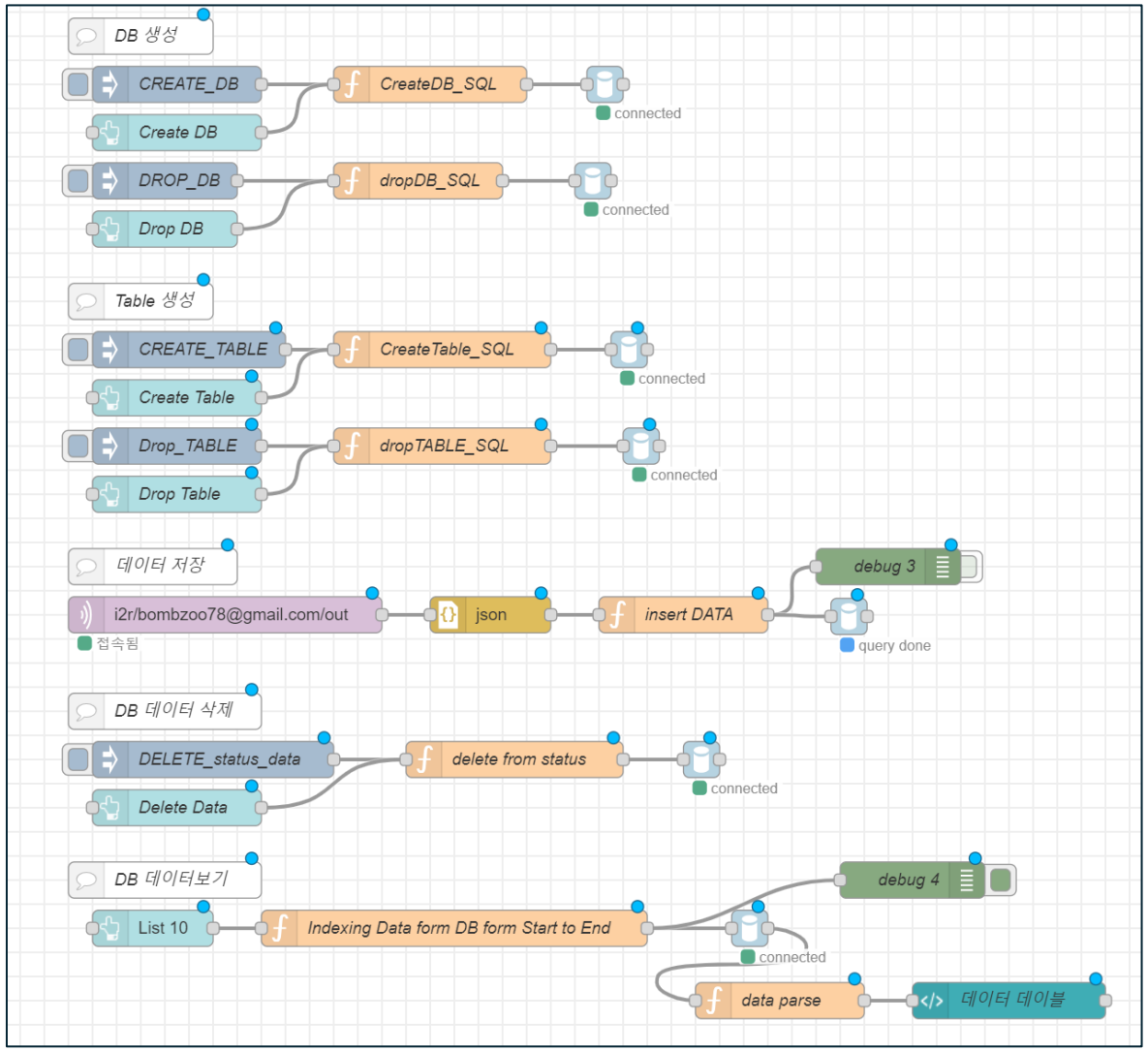
```
"timestamp " :1727170478,
"email": "bombzoo78@gmail.com",
"mac":"EC:64:C9:43:E8:B8",
"temp":28.4,
"humi":38,
"in":[0,0,0,0],
"out":[0,0,0,0]
```



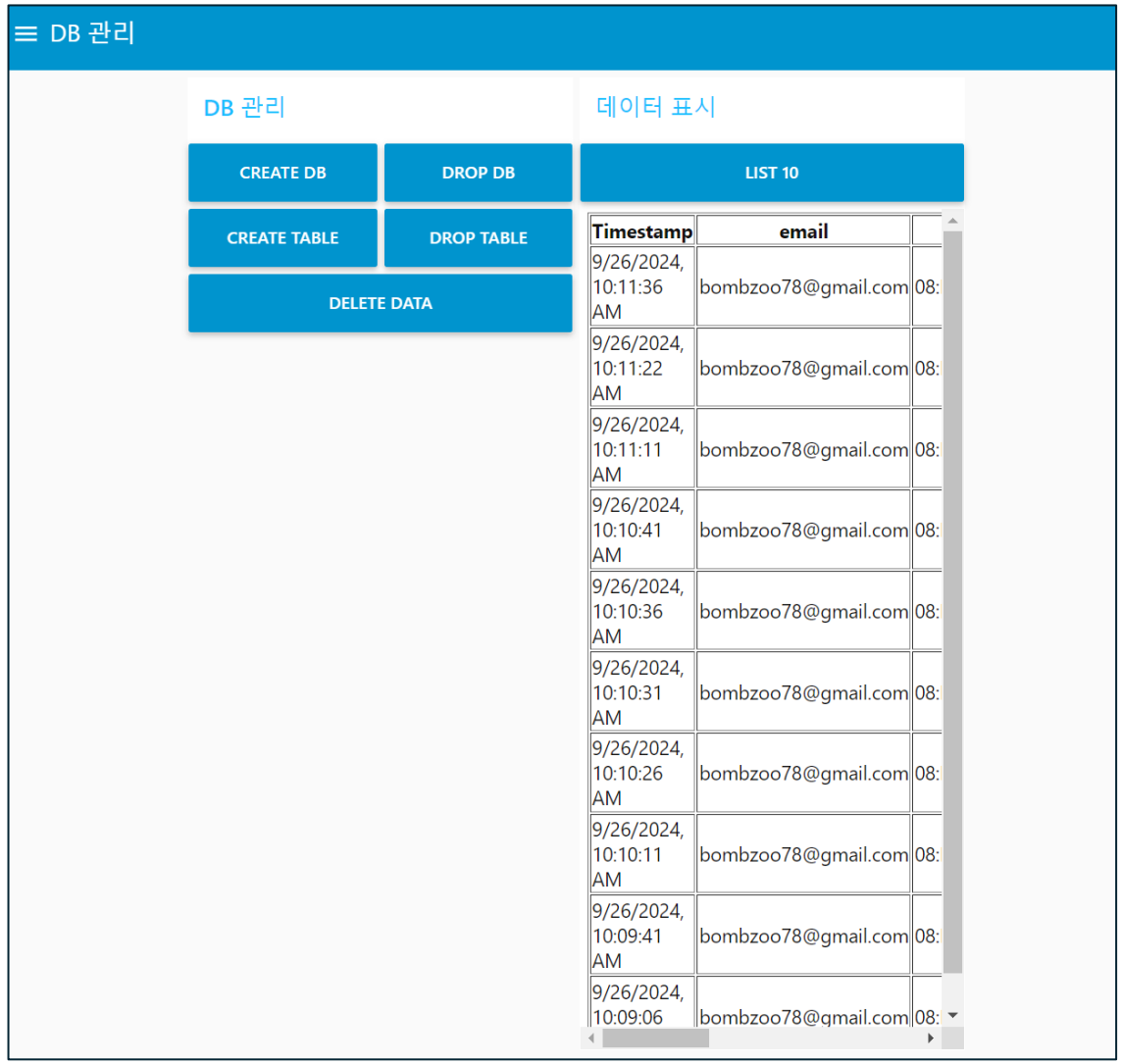

```
// 메시지로부터 JSON 데이터를 추출합니다.
const jsonData = msg.payload;

// const timestamp = new Date().toLocaleString('ko-KR', { timeZone: 'Asia/Seoul', hour12: false });
const timestamp = Math.floor(new Date().getTime() / 1000);
const email = jsonData.email;
const mac = jsonData.mac;
const temp = jsonData.temp;
const humi = jsonData.humi;
const in00 = jsonData.in[0];
const in01 = jsonData.in[1];
const in02 = jsonData.in[2];
const in03 = jsonData.in[3];
const out00 = jsonData.out[0];
const out01 = jsonData.out[1];
const out02 = jsonData.out[2];
const out03 = jsonData.out[3];

// MySQL 쿼리를 생성합니다. 문자열 값은 작은따옴표로 감싸야 하고, 숫자 값은 그대로 넣습니다.
let sql = `INSERT INTO iot.status (timestamp, email, mac, temp, humi, in00, in01, in02, in03, out00, out01, out02, out03)
VALUES ('${timestamp}', '${email}', '${mac}', ${temp}, ${humi}, ${in00}, ${in01}, ${in02}, ${in03}, ${out00}, ${out01},
${out02}, ${out03});`;

// 메시지에 SQL 쿼리 추가
msg.topic = sql;

// 메시지 전달
return msg;
```

Node-Red 사용법

DB 관리



Dashboard



Timestamp	email	
9/26/2024, 10:11:36 AM	bombzoo78@gmail.com	08:
9/26/2024, 10:11:22 AM	bombzoo78@gmail.com	08:
9/26/2024, 10:11:11 AM	bombzoo78@gmail.com	08:
9/26/2024, 10:10:41 AM	bombzoo78@gmail.com	08:
9/26/2024, 10:10:36 AM	bombzoo78@gmail.com	08:
9/26/2024, 10:10:31 AM	bombzoo78@gmail.com	08:
9/26/2024, 10:10:26 AM	bombzoo78@gmail.com	08:
9/26/2024, 10:10:11 AM	bombzoo78@gmail.com	08:
9/26/2024, 10:09:41 AM	bombzoo78@gmail.com	08:
9/26/2024, 10:09:06	bombzoo78@gmail.com	08: