# ✅ Step-by-Step: Migrate from RDS to Local PostgreSQL on EC2

---

## 1. Install PostgreSQL on EC2 (Outside Docker)

Login to EC2 and install PostgreSQL:

bash
CopyEdit

```
sudo apt update
sudo apt install postgresql postgresql-contrib -y
```

Enable and start the service:

bash
CopyEdit

```
sudo systemctl enable postgresql
sudo systemctl start postgresql
```

---

## 2. Create a New PostgreSQL Database Locally

Switch to the `postgres` user:

bash
CopyEdit

```
sudo -i -u postgres
```

Create DB and user:

bash
CopyEdit

```
psql
CREATE DATABASE myappdb;
CREATE USER myuser WITH PASSWORD 'mypassword';
GRANT ALL PRIVILEGES ON DATABASE myappdb TO myuser;
\q
```

```
exit
```

✅ Now your local PostgreSQL is ready.

---

### 3. Dump Your Data from AWS RDS

From EC2 (or anywhere with access to RDS), run:

bash
CopyEdit
```
pg_dump -h <rds-hostname> -U <rds-username> -d <rds-dbname> -Fc -f rds_backup.dump
```

You can install `postgresql-client` on EC2 if `pg_dump` is missing:

bash
CopyEdit
```
sudo apt install postgresql-client
```

---

### 4. Restore Dump to Local PostgreSQL

Run this from EC2:

bash
CopyEdit
```
pg_restore -U myuser -d myappdb -v rds_backup.dump
```

> You might need `PGPASSWORD=mypassword` before the command if `pg_restore` asks for a password:

bash
CopyEdit
```
PGPASSWORD=mypassword pg_restore -U myuser -d myappdb -v rds_backup.dump
```

---

## 5. Update Your Backend Container to Use Local DB

**Update your backend's `.env` or config:**

Instead of:

bash
CopyEdit
```
DATABASE_URL=postgres://user:password@rds-host:5432/dbname
```

Use:

bash
CopyEdit
```
DATABASE_URL=postgres://myuser:mypassword@host.docker.internal:5432/myappdb
```

OR if `host.docker.internal` doesn't work on Linux (Docker on EC2), use the host IP:

bash
CopyEdit
```
DATABASE_URL=postgres://myuser:mypassword@172.17.0.1:5432/myappdb
```

**Alternatively**, use the internal IP of EC2 itself:

bash
CopyEdit
```
DATABASE_URL=postgres://myuser:mypassword@<your-ec2-private-ip>:5432/myappdb
```

---

## 6. Rebuild Backend Container with New DB Config

bash
CopyEdit
```
docker-compose down
docker-compose up --build -d
```

Or however you're starting your backend container — just make sure it uses the updated `.env`.

---

### 7. Verify Everything

- Check backend logs: `docker logs <backend-container>`
- Try hitting API routes that use DB

Check PostgreSQL logs (optional):

bash
CopyEdit

```
sudo journalctl -u postgresql
```

- 

---

### 🛡️ Optional: Harden Local PostgreSQL for Security

Edit PostgreSQL config:

bash
CopyEdit

```
sudo nano /etc/postgresql/14/main/pg_hba.conf
```

- 
    - Use `md5` instead of `trust`
- Open port `5432` only if containers are on different bridge networks.
- Use a firewall (`ufw`) if needed.

---

## 🧠 Summary

| Step | Description |
|---|---|
| 1 | Install PostgreSQL on EC2 |
| 2 | Create DB and user |

| | |
|---|---|
| 3 | Dump from RDS using `pg_dump` |
| 4 | Restore to EC2 PostgreSQL with `pg_restore` |
| 5 | Update `.env` in backend container |
| 6 | Rebuild containers |
| 7 | Test and secure your setup |