Quick Fix Commands for ScalarDB Connection Issues

For Linux/Mac Users:

Save the bash script as `reconnect.sh` and run:

```bash
chmod +x reconnect.sh
./reconnect.sh
```

For Windows Users:

Save the PowerShell script as `reconnect.ps1` and run:

```powershell
Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser
.\reconnect.ps1
```

One-Liner Quick Fixes:

1. Quick PostgreSQL + Dashboard Start (Linux/Mac):

```bash
docker run -d --name scalardb-postgres -e POSTGRES_DB=scalardb -e POSTGRES_USER=postgres -e POSTGR
```

2. Quick PostgreSQL + Dashboard Start (Windows):

```cmd
docker run -d --name scalardb-postgres -e POSTGRES_DB=scalardb -e POSTGRES_USER=postgres -e POSTGR
```

3. Stop Everything and Restart (Linux/Mac):

```bash
docker stop scalardb-postgres; docker rm scalardb-postgres; pkill -f "node server.js"; pkill -f "node medical-serve
```

4. Stop Everything and Restart (Windows):

powershell

```
docker stop scalardb-postgres; docker rm scalardb-postgres; Get-Process node | Stop-Process -Force
```

Manual Step-by-Step Fix:

Step 1: Fix PostgreSQL

bash

```
# Stop existing container
docker stop scalardb-postgres 2>/dev/null || true
docker rm scalardb-postgres 2>/dev/null || true

# Start fresh PostgreSQL container
docker run -d \
  --name scalardb-postgres \
  -e POSTGRES_DB=scalardb \
  -e POSTGRES_USER=postgres \
  -e POSTGRES_PASSWORD=postgres \
  -p 5432:5432 \
  postgres:15

# Wait for it to start
sleep 10

# Test connection
docker exec scalardb-postgres pg_isready -U postgres
```

Step 2: Create Sample Tables

bash

```
docker exec -i scalardb-postgres psql -U postgres -d scalardb << 'EOF'
CREATE TABLE IF NOT EXISTS sample_customer (
  customer_id SERIAL PRIMARY KEY,
  name VARCHAR(255) NOT NULL,
  credit_limit INTEGER DEFAULT 10000,
  credit_total INTEGER DEFAULT 0
);

INSERT INTO sample_customer (name, credit_limit, credit_total)
VALUES
  ('Yamada Taro', 10000, 0),
  ('Yamada Hanako', 10000, 0),
  ('Suzuki Ichiro', 10000, 0)
ON CONFLICT DO NOTHING;

CREATE TABLE IF NOT EXISTS sample_order (
  order_id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  customer_id INTEGER REFERENCES sample_customer(customer_id),
  timestamp BIGINT DEFAULT EXTRACT(epoch FROM NOW()) * 1000,
  total_amount INTEGER DEFAULT 0
);
EOF
```

Step 3: Fix Node.js Dependencies

bash

```bash
# Create package.json if missing
cat > package.json << 'EOF'
{
  "name": "scalardb-dashboard",
  "version": "1.0.0",
  "main": "server.js",
  "dependencies": {
    "express": "^4.18.2",
    "pg": "^8.11.3",
    "cors": "^2.8.5"
  }
}
EOF

# Install dependencies
npm install
```

Step 4: Start Services

bash

```bash
# Kill existing Node processes
pkill -f "node server.js" || true
pkill -f "node medical-server.js" || true

# Start dashboard
nohup node server.js > dashboard.log 2>&1 &

# Start medical app (if exists)
if [ -f "medical-server.js" ]; then
    nohup node medical-server.js > medical.log 2>&1 &
fi

# Wait and test
sleep 5
curl http://localhost:3000/api/health
curl http://localhost:3001/api/health
```

Troubleshooting Commands:

Check what's running on ports:

bash

```
# Linux/Mac
lsof -i :3000
lsof -i :3001
lsof -i :5432

# Windows
netstat -ano | findstr :3000
netstat -ano | findstr :3001
netstat -ano | findstr :5432
```

Check Docker containers:

bash

```
docker ps -a
docker logs scalardb-postgres
```

Check Node.js processes:

bash

```
# Linux/Mac
ps aux | grep node

# Windows
tasklist | findstr node
```

View logs:

bash

```
tail -f dashboard.log
tail -f medical.log
docker logs scalardb-postgres
```

Expected Results:

After running the fix script, you should see:

- ✅ PostgreSQL: Connected
- ✅ Test Dashboard: Running on port 3000

- ✅ Medical App: Running on port 3001

Your applications should be accessible at:

- http://localhost:3000 (Dashboard)
- http://localhost:3001 (Medical App)