FinalFunnel - Complete Deployment Guide

📋 System Overview

This system is designed to handle **1TB+ data** with:

- **Sub-second query response** using materialized views and advanced indexing
- **Bulk upload processing** at 1000-5000 rows/second
- **PostgreSQL 15** with performance optimization
- **Redis caching** for ultra-fast filter options
- **Automated backups** and monitoring

🖥️ Hardware Requirements

Minimum (Testing)

- **CPU:** 4 cores
- **RAM:** 8 GB
- **Storage:** 100 GB SSD
- **Network:** 100 Mbps

Recommended (Production)

- **CPU:** 8+ cores
- **RAM:** 32 GB
- **Storage:** 500 GB NVMe SSD
- **Network:** 1 Gbps
- **Backup Storage:** 1 TB

Large Scale (1TB+ data)

- **CPU:** 16+ cores
- **RAM:** 64 GB+
- **Storage:** 2 TB NVMe SSD (RAID 10)
- **Network:** 10 Gbps
- **Backup Storage:** 5 TB

🚀 Quick Start Deployment

Step 1: Prepare Ubuntu Server (20.04 LTS or 22.04 LTS)

```bash
# Update system
sudo apt update && sudo apt upgrade -y

# Download setup script
wget https://raw.githubusercontent.com/YOUR_REPO/finalfunnel/main/setup_system.sh

# Make executable
chmod +x setup_system.sh

# Run setup (this will take 10-15 minutes)
sudo bash setup_system.sh
```

Step 2: Configure Environment

```bash
cd /home/ubuntu/finalfunnel

# Copy template and edit
cp .env.template .env
nano .env
```

Edit these critical values:

```env
# Database - CHANGE PASSWORD!
DB_PASSWORD=YOUR_SECURE_DB_PASSWORD_HERE

# Application - CHANGE PASSWORD!
APP_USERNAME=admin
APP_PASSWORD=YOUR_SECURE_ADMIN_PASSWORD_HERE

# Redis (optional - keep default if running locally)
REDIS_HOST=localhost
```

Step 3: Initialize Database

```bash
# Activate virtual environment
source venv/bin/activate

# Run database initialization
python init_database.py
```

Expected output:

```
Creating optimized database schema...
✅ Database initialized successfully!
📊 Tables: 0 rows in fact_contacts, 0 in fact_companies
🔍 Indexes: 45 total indexes created
```

Step 4: Start Application

```bash
# Using supervisor (recommended)
sudo supervisorctl start finalfunnel

# OR manual start for testing
streamlit run 0_Home.py --server.port 8501
```

Step 5: Access Application

Open your browser:

- **Local:** http://localhost:8501
- **Remote:** http://your-server-ip:8501

Default Login:

- Username: (from .env APP_USERNAME)
- Password: (from .env APP_PASSWORD)

🔒 Security Setup

1. Change Default Passwords

PostgreSQL:

```bash
sudo -u postgres psql
ALTER USER finalfunnel_user WITH PASSWORD 'your_new_strong_password';
\q
```

Update `.env` with new password.

2. Setup SSL with Let's Encrypt

```bash
# Edit nginx config first
sudo nano /etc/nginx/sites-available/finalfunnel

# Change: server_name your-domain.com;

# Get SSL certificate
sudo certbot --nginx -d your-domain.com

# Auto-renewal is setup automatically
```

3. Firewall Configuration

```bash
# Allow SSH, HTTP, HTTPS
sudo ufw allow 22/tcp
sudo ufw allow 80/tcp
sudo ufw allow 443/tcp

# Block direct access to Streamlit port
sudo ufw deny 8501/tcp

# Enable firewall
sudo ufw enable
```

4. Database Security

```bash
# Edit PostgreSQL config
sudo nano /etc/postgresql/15/main/pg_hba.conf

# Change this line:
# local   all   all   peer
# TO:
# local   all   all   md5

# Restart PostgreSQL
sudo systemctl restart postgresql
```

📤 How to Upload Data

Method 1: Web Interface (Up to 500MB)

1. Navigate to **Upload Data** page
2. Click **Download Template**
3. Fill template with your data
4. Click **Upload Your Data**
5. Select file and click **Validate**
6. If validation passes, click **Process Upload**

Method 2: Direct Database Import (500MB+)

For very large files:

```bash
# 1. Copy file to server
scp your_large_file.csv ubuntu@your-server:/tmp/

# 2. SSH to server
ssh ubuntu@your-server

# 3. Run import script
cd /home/ubuntu/finalfunnel
source venv/bin/activate
python scripts/bulk_import.py /tmp/your_large_file.csv
```

Data Quality Requirements

Required Columns (exact order):

1. comp_name
2. comp_domain
3. annrev
4. comp_industry
5. comp_linkedin
6. firstname
7. lastname
8. jobtitle
9. manlevel
10. empemail (REQUIRED - must be unique)
11. emplinkedin
12. country_code
13. comp_phone
14. comp_street
15. comp_city
16. comp_state
17. comp_country
18. comp_zipcode
19. qa_disposition
20. empsize

Data Formats:

- **empemail:** Must be valid email (required)
- **annrev:** "10M", "5B", or number
- **empsize:** "50-200", "1000+" or number
- **URLs:** No http://, https://, or www. needed
- **Phone:** Any format accepted

🔍 Fast Filtering & Performance

Understanding the Architecture

The system uses a **3-tier data structure**:

1. **Fact Tables** (fact_contacts, fact_companies)
   - Normalized storage
   - Optimized for writes
   - Foreign key relationships
2. **Materialized View** (mv_contacts_data)
   - Pre-joined denormalized data
   - Optimized for reads
   - Refreshed after uploads
3. **Indexes** (45+ specialized indexes)
   - B-tree for exact matches
   - GIN for text search
   - Composite for filter combinations

Query Performance

Expected response times (after warm-up):

Records	Filters	Response Time
1M	0-2	<100ms
10M	0-2	<200ms
100M	0-2	<500ms
1B	0-2	<2s
1M	5+	<50ms
100M	5+	<300ms

Performance Optimization Tips

1. **Use multiple filters** - More filters = faster queries (counter-intuitive but true due to indexes)
2. **Refresh materialized view after bulk uploads:**

```sql
REFRESH MATERIALIZED VIEW CONCURRENTLY mv_contacts_data;
```

3. **Update statistics weekly:**

```sql
ANALYZE mv_contacts_data;
```

4. **Monitor slow queries:**

```sql
SELECT * FROM pg_stat_statements
WHERE mean_exec_time > 1000
ORDER BY mean_exec_time DESC;
```

🔧 Maintenance

Daily Tasks (Automated)

- Cache refresh (Redis)
- Backup validation
- Log rotation

Weekly Tasks

```bash
# Update statistics
sudo -u postgres psql -d finalfunnel_db -c "VACUUM ANALYZE;"

# Check index usage
sudo -u postgres psql -d finalfunnel_db -c "SELECT * FROM v_index_usage WHERE idx_scan < 100;"
```

Monthly Tasks

```bash
# Full system backup
sudo -u postgres pg_dump finalfunnel_db > backup_$(date +%Y%m%d).sql

# Compress
gzip backup_$(date +%Y%m%d).sql

# Upload to S3 or backup storage
```

Database Size Monitoring

sql

```sql
-- Check table sizes
SELECT * FROM v_table_sizes;

-- Check growth
SELECT
    schemaname,
    tablename,
    pg_size_pretty(pg_total_relation_size(schemaname||'.'||tablename)) AS size,
    pg_size_pretty(pg_total_relation_size(schemaname||'.'||tablename) -
                   pg_relation_size(schemaname||'.'||tablename)) AS index_size
FROM pg_tables
WHERE schemaname = 'public'
ORDER BY pg_total_relation_size(schemaname||'.'||tablename) DESC
LIMIT 10;
```

🐛 Troubleshooting

Application Won't Start

bash

```bash
# Check logs
tail -f /var/log/finalfunnel.err.log
tail -f /var/log/finalfunnel.out.log

# Check supervisor status
sudo supervisorctl status finalfunnel

# Restart
sudo supervisorctl restart finalfunnel
```

Slow Queries

bash

```
# Check active queries
sudo -u postgres psql -d finalfunnel_db -c "
SELECT pid, now() - query_start as duration, query
FROM pg_stat_activity
WHERE state = 'active' AND now() - query_start > interval '5 seconds';
"

# Kill slow query
sudo -u postgres psql -d finalfunnel_db -c "SELECT pg_terminate_backend(PID_HERE);"
```

Database Connection Issues

bash

```
# Check PostgreSQL status
sudo systemctl status postgresql

# Check connections
sudo -u postgres psql -c "SELECT count(*) FROM pg_stat_activity;"

# Increase max connections (if needed)
sudo nano /etc/postgresql/15/main/postgresql.conf
# Change: max_connections = 200
sudo systemctl restart postgresql
```

Out of Disk Space

bash

```
# Check disk usage
df -h

# Find large files
du -sh /var/lib/postgresql/15/main/*

# Clean old backups
find /home/ubuntu/backups -name "*.sql.gz" -mtime +30 -delete

# Vacuum database
sudo -u postgres psql -d finalfunnel_db -c "VACUUM FULL;"
```

📊 Monitoring & Alerts

Setup Monitoring Dashboard

```bash
# Install pgAdmin 4 (optional)
curl https://www.pgadmin.org/static/packages_pgadmin_org.pub | sudo apt-key add
sudo sh -c 'echo "deb https://ftp.postgresql.org/pub/pgadmin/pgadmin4/apt/$(lsb_release -cs) pgadmin4 main" > /etc/apt/sourc
sudo apt update
sudo apt install pgadmin4
```

Key Metrics to Monitor

1. **Database Size Growth**
2. **Query Response Time**
3. **Active Connections**
4. **CPU & Memory Usage**
5. **Disk I/O**
6. **Upload Success Rate**

🔄 Backup & Recovery

Automated Backup Script

Create `/home/ubuntu/finalfunnel/scripts/backup.sh`:

```bash
#!/bin/bash
BACKUP_DIR="/home/ubuntu/backups"
DATE=$(date +%Y%m%d_%H%M%S)
DAYS_TO_KEEP=30

# Create backup
sudo -u postgres pg_dump finalfunnel_db | gzip > $BACKUP_DIR/backup_$DATE.sql.gz

# Upload to S3 (optional)
# aws s3 cp $BACKUP_DIR/backup_$DATE.sql.gz s3://your-bucket/backups/

# Delete old backups
find $BACKUP_DIR -name "backup_*.sql.gz" -mtime +$DAYS_TO_KEEP -delete

echo "Backup completed: backup_$DATE.sql.gz"
```

Setup cron:

```bash
crontab -e

# Add: Daily backup at 2 AM
0 2 * * * /home/ubuntu/finalfunnel/scripts/backup.sh >> /var/log/backup.log 2>&1
```

Recovery Process

```bash
# Stop application
sudo supervisorctl stop finalfunnel

# Drop and recreate database
sudo -u postgres psql -c "DROP DATABASE finalfunnel_db;"
sudo -u postgres psql -c "CREATE DATABASE finalfunnel_db;"

# Restore from backup
gunzip -c backup_20240115_020000.sql.gz | sudo -u postgres psql finalfunnel_db

# Restart application
sudo supervisorctl start finalfunnel
```

📞 Support & Resources

Log Locations

- Application: `/var/log/finalfunnel.out.log`
- Errors: `/var/log/finalfunnel.err.log`
- PostgreSQL: `/var/log/postgresql/postgresql-15-main.log`
- Nginx: `/var/log/nginx/access.log` & `error.log`

Useful Commands

```bash
# Application status
sudo supervisorctl status finalfunnel

# Database size
sudo -u postgres psql -d finalfunnel_db -c "SELECT pg_size_pretty(pg_database_size('finalfunnel_db'));"

# Active users
sudo -u postgres psql -d finalfunnel_db -c "SELECT count(*) FROM pg_stat_activity WHERE datname='finalfunnel_db';"

# Clear cache
redis-cli FLUSHALL

# Restart everything
sudo supervisorctl restart finalfunnel
sudo systemctl restart postgresql
sudo systemctl restart redis-server
sudo systemctl restart nginx
```

🎯 Best Practices

1. **Regular Backups** - Automate daily backups
2. **Monitor Performance** - Setup alerts for slow queries
3. **Update Statistics** - Weekly ANALYZE
4. **Security Updates** - Monthly OS updates
5. **Test Restores** - Quarterly backup restoration tests
6. **Capacity Planning** - Monitor growth trends
7. **Documentation** - Keep deployment notes
8. **Access Control** - Limit database access
9. **SSL/TLS** - Always use HTTPS in production
10. **Change Passwords** - Never use defaults

✅ Post-Deployment Checklist

- [ ] Ubuntu server updated
- [ ] PostgreSQL installed and configured
- [ ] Database initialized with schema
- [ ] Application running via supervisor
- [ ] Nginx configured with SSL
- [ ] Firewall rules applied
- [ ] Passwords changed from defaults
- [ ] Automated backups setup
- [ ] Monitoring configured
- [ ] Test upload completed
- [ ] Filter performance verified
- [ ] Documentation reviewed
- [ ] Admin access confirmed
- [ ] Support contacts saved

System Version: 1.0.0
Last Updated: November 2024
Designed for: 1TB+ data with sub-second response times