Database Backup and Restore Procedure

Project Deliverable

Course B1AI

Class:

Author: François LANGE

Date: 2026/06/19

Table of Contents

The current version is #ident "@(#)$Format:LocalFoodAI_lanfr144:generate_docs.py:Francois Lange:lanfr144@school.lu:2026/06/16 21:48:22:Francois Lange:lanfr144@school.lu:2026/06/16 21:48:22:2a8ed056889f3b796f9266feda591b12b72f3b96:HEAD -> main, origin/main:$"

Database Backup and Restore Procedure

1. Overview & Policy

To guarantee clinical records integrity and high availability, Local Food AI enforces a strict backup schedule.

- **Scope**: Includes MySQL schemas (`food_db`), user profiles (`app_auth`), and configuration states.
- **Retention Plan**: Automated daily backups with a strict 7-day rolling window purge.
- **Storage Location**: Stored securely inside the persistent `/backups` directory on the host server.

---

2. Automated Daily Backups

The automated backup mechanism runs via a host cron job pointing to `backup_db.sh`.

- The script dynamically detects the active MySQL container name (`food-mysql-1` or `food_project-mysql-1`).
- It executes `mysqldump` directly inside the container without exposing root passwords to shell logs.
- Outputs are compressed via `gzip` and timestamped: `food_db_YYYYMMDD_HHMM.sql.gz`.

Cron Configuration Example:

To run the backup daily at 02:00 AM, add the following to `/etc/crontab`:

```
0 2 * * * root /bin/bash /c/Users/lanfr144/Documents/DOPRO1/Antigravity/Food/backup_db.sh >> /var/log/backup_db.log 2>&1
```

---

3. Manual Backup Execution

If a system migration or major upgrade is scheduled, perform a manual dump using the following command:

```
# 1. Navigate to the project directory
cd /c/Users/lanfr144/Documents/DOPRO1/Antigravity/Food

# 2. Run the backup wrapper
bash backup_db.sh
```

Verify the output exists inside the backups folder:

```
ls -lh backups/
```

---

4. Step-by-Step Restore Procedure

In the event of database corruption or hardware failure, follow these exact steps to restore the database.

Step 4.1: Identify the Target Backup File

List available files and pick the desired timestamp:

```
ls -la backups/
# Example Target: backups/food_db_20260521_1100.sql.gz
```

Step 4.2: Verify MySQL Container Health

Ensure the MySQL service container is running and healthy:

```
docker ps --filter name=mysql
```

Step 4.3: Execute Restore Stream

Decompress the backup on-the-fly and pipe it directly into the running MySQL container:

```
# Adjust the container name ('food-mysql-1' or 'food_project-mysql-1') based on
active deployment
gunzip < backups/food_db_20260521_1100.sql.gz | docker exec -i food-mysql-1 mysql
-u root -proot_pass food_db
```

Step 4.4: Verify Restored Tables

Log in to the database and query the core table to confirm the tables are intact and populated:

```
docker exec -it food-mysql-1 mysql -u food_reader -preader_pass food_db -e "SELECT
COUNT(*) FROM products_core;"
```

Expected result: A count of OpenFoodFacts entries (typically > 10,000 records).

---

5. Verification & Health Check Loops

Operators must verify the backup archive integrity weekly:

- Copy the `.gz` backup to a local testing workspace.
- Run `gzip -t backups/filename.sql.gz` to ensure the archive is not corrupted.
- Test restoring to a local fallback container instance to verify data accessibility.

References

- **OpenFoodFacts Dataset & API Catalog**: Detailed food ingredients database. (https://world.openfoodfacts.org/)
- **Ollama Local LLM Inference Engine**: Lightweight instruction-following llama3.2 runtimes. (https://ollama.com/)
- **Zabbix Enterprise Telemetry and Monitoring**: System health and performance logging. (https://www.zabbix.com/)

Index