1. MySQL Backup & Restore

Backup Script (backup_mysql.sh)

```
#!/bin/bash
DB_NAME="your_mysql_db"
BACKUP_DIR="/backups/mysql"
TIMESTAMP=$(date +'%Y%m%d_%H%M')
BACKUP_FILE="${BACKUP_DIR}/${DB_NAME}_${TIMESTAMP}.sql.gz"

mkdir -p "${BACKUP_DIR}"
mysqldump -u root -p "${DB_NAME}" | gzip > "${BACKUP_FILE}"

echo "Backup saved to ${BACKUP_FILE}"
# Rotate backups older than 14 days
find "${BACKUP_DIR}" -name "*.sql.gz" -mtime +14 -delete
```

- **mysqldump** creates logical SQL dumps, then **gzip**

Restore Script (restore_mysql.sh)

```
#!/bin/bash
DB_NAME="your_mysql_db"
BACKUP_FILE="$1"  # Path to .sql.gz file

if [ -z "${BACKUP_FILE}" ]; then
  echo "Usage: restore_mysql.sh path/to/backup.sql.gz"
  exit 1
fi

gunzip -c "${BACKUP_FILE}" | mysql -u root -p "${DB_NAME}"

echo "Database ${DB_NAME} restored from ${BACKUP_FILE}"
```

2. PostgreSQL Backup & Restore

Backup Script (backup_postgres.sh)

bash

CopyEdit

```
#!/bin/bash
DB_NAME="your_pg_db"
BACKUP_DIR="/backups/postgres"
TIMESTAMP=$(date +'%Y%m%d_%H%M')
BACKUP_FILE="${BACKUP_DIR}/${DB_NAME}_${TIMESTAMP}.sql.gz"
USER="postgres"

mkdir -p "${BACKUP_DIR}"
pg_dump -U ${USER} -F p "${DB_NAME}" | gzip > "${BACKUP_FILE}"

echo "Backup saved to ${BACKUP_FILE}"
find "${BACKUP_DIR}" -name "*.sql.gz" -mtime +14 -delete
```

Restore Script (restore_postgres.sh)

bash

CopyEdit

```
#!/bin/bash
DB_NAME="your_pg_db"
BACKUP_FILE="$1"
USER="postgres"

if [ -z "${BACKUP_FILE}" ]; then
  echo "Usage: restore_postgres.sh path/to/backup.sql.gz"
  exit 1
fi
```

gunzip -c "${BACKUP_FILE}" | psql -U ${USER} -d "${DB_NAME}"

echo "Database ${DB_NAME} restored from ${BACKUP_FILE}"

- **Uses psql < dump.sql for plain-format**

Documentation Outline

1. **Overview**
    - **Purpose: Recover databases after system failure, accidental deletes, etc.**
2. **Script Descriptions**
    - **backup_mysql.sh, restore_mysql.sh**
    - **backup_postgres.sh, restore_postgres.sh**
3. **Execution Instructions**
    - **Grant executable permissions: chmod +x backup_*.sh restore_*.sh**
    - **Run backups (e.g., daily via cron): crontab -e and schedule.**
4. **Data Integrity & Validation**
    - **Compressed backups prevent data corruption.**
    - **Post-restore, verify by:**

sql

CopyEdit

SELECT COUNT(*) FROM important_table;

    - **Optional: checksum original vs restored data.**

5. **Retention Policies**
    - **Scripts auto-delete backups older than 14 days using find -mtime +14.**
6. **Failure Handling**
    - **If restore fails, error bubbles up. Log failures via || echo "Restore failed" >> /var/log/db_restore.log.**
7. **Post-Restore Verification**

- **Check successful psql/mysql exit codes.**
- **Sample queries to ensure data presence.**

Final Deliverables

- **Scripts:**
  - **backup_mysql.sh, restore_mysql.sh**
  - **backup_postgres.sh, restore_postgres.sh**
- **Documentation:**
  - **Purpose and scope**
  - **Execution steps**
  - **Data integrity & retention strategy**
  - **Failure handling**
  - **Verification procedures**