MyWebUniversity.com

Introduction to SQL

& Databases

MySQL · Oracle · MS SQL Server · MS Access · PostgreSQL

A Complete Beginner-to-Pro Video Course

youtube.com/@MyWebUniversity

youtube.com/@MyWebUniversity-Dari

What You Will Learn Today

60+ topics across 12 sections

01 What is a Database & Why It Matters
02 Types of Databases (Relational, OO, Hierarchical…)
03 DBMS Overview · MySQL · Oracle · MS SQL · PostgreSQL · MS Access, LDAP
04 DB Architecture: SGA, Buffer Pool, Tablespace, Daemons
05 Keys: PK · FK · SK · AK · CK · Surrogate
06 Tablespaces, Indices, Tables, Records, Fields, Rows

07 Normalization (1NF → 3NF, & BCNF) & Relationships (1:1, 1:M, M:N)
08 DDL — CREATE, DROP, ALTER
09 DML — INSERT, UPDATE, SELECT, DELETE
10 DCL — GRANT & REVOKE
11 Distributed vs Centralized · Replication · Snapshots
12 Data Science · Python/Java/C++ · ODBC · API · Careers

SECTION 01

What is a Database?

The foundation of all modern software systems

What is a Database?

Organized collection of structured information stored electronically

Simple Definition

A database is an organized collection of data stored and accessed electronically — like a super-powered, structured filing system.

Real-Life Analogy

Think of your smartphone contacts: each contact has a name, phone, email. That is a simple database — structured, searchable, updatable.

Why Databases Matter

- Every website you visit uses a database
- Your bank account, health records, shopping cart
- Facebook: 2.9B users — all stored in databases
- Google indices 100B+ web pages in databases
- Without databases, modern software does not exist

Real-World Database Examples

Every industry runs on databases

Banking

JPMorgan, Bank of America — Oracle/DB2 stores billions of transactions daily. 24/7 ACID-compliant.

Healthcare

Mayo Clinic, NHS — patient records, prescriptions, lab results in Oracle/MS SQL Server.

E-Commerce

Amazon, eBay — MySQL + Aurora store product catalogs, orders, inventory in real time.

Social Media

Facebook (MySQL), Twitter (PostgreSQL) — billions of rows, billions of users worldwide.

Finance & Trading

NYSE, NASDAQ — ultra-low-latency databases process millions of stock trades per second.

Cloud & SaaS

Salesforce, SAP — PostgreSQL/Oracle power CRM, ERP, and analytics for Fortune 500 companies.

SECTION 02

Types of Databases

Relational · Object-Oriented · Hierarchical · NoSQL & More

Types of Databases

Choosing the right type is critical for your application

Relational (RDBMS)

Data in tables with rows/columns. Uses SQL. Examples: MySQL, Oracle, PostgreSQL, MS SQL, MS Access. Best for: structured business data.

Object-Oriented (OODBMS)

Stores data as objects (like OOP). Inherits classes, polymorphism. Examples: db4o, ObjectDB. Best for: CAD, multimedia, complex objects.

Hierarchical (Tree)

Parent→Child tree structure. Examples: IBM IMS, Windows Registry, LDAP Directory Services. Best for: org charts, file systems.

Network DB

Like hierarchical but many-to-many links. Examples: IDMS (CA). Mostly legacy mainframe systems.

NoSQL / Document

Schema-less, JSON/BSON docs. Examples: MongoDB, CouchDB, Cassandra, Redis. Best for: big data, real-time web apps.

NewSQL / Cloud-Native

Scalable relational DBs. Examples: Google Spanner, CockroachDB, Amazon Aurora. Best for: global scale + ACID.

Hierarchical Database — LDAP Example

Lightweight Directory Access Protocol — used by Active Directory, OpenLDAP

LDAP Tree Structure:

```
dc=mywebuniversity,dc=com   (Root)
  └─ ou=People
       ├─ cn=Wahidullah   (uid, email, password)
       └─ cn=Student01
  └─ ou=Groups
       └─ cn=Admins
```

- Used by Windows Active Directory for user/group management
- Each entry has a Distinguished Name (DN)
- Parent-child: one parent, many children — tree traversal
- Protocol: LDAP runs on TCP port 389, LDAPS on 636
- Commands: ldapsearch, ldapadd, ldapmodify, ldapdelete

SECTION 03

DBMS Overview

MySQL · Oracle · MS SQL Server · MS Access · PostgreSQL

What is a DBMS?

Database Management System — the software that manages your database

DBMS Definition

A DBMS is software that enables users and applications to create, read, update, and delete data in a database. It handles storage, retrieval, security, backup, and concurrency.

Core DBMS Functions

- Data Definition — create/modify schema (tables, views)
- Data Manipulation — CRUD operations
- Access Control — user permissions, roles
- Transaction Management — ACID properties
- Backup & Recovery — protect against data loss
- Concurrency Control — multiple users, no conflicts

ACID Properties

A — Atomicity
All or nothing. Either all changes commit or all roll back.

C — Consistency
DB moves from one valid state to another.

I — Isolation
Transactions don't interfere with each other.

D — Durability
Committed data survives crashes.

Major DBMS Platforms Compared

MySQL · Oracle · MS SQL Server · MS Access · PostgreSQL

DBMS	Owner	License	Best For	Used By
MySQL	Oracle / Community	Open Source / Commercial	Web apps, startups	Facebook, YouTube, Airbnb
Oracle DB	Oracle Corp	Commercial	Enterprise, banking	Banks, airlines, gov
MS SQL	Microsoft	Commercial	Windows enterprise	Healthcare, insurance
MS Access	Microsoft	Office 365	Small/desktop DBs	Small business, education
PostgreSQL	Open Source	Free (BSD)	Advanced analytics, GIS	Apple, USPS, Instagram

SECTION 04

DB Architecture

SGA · Buffer Pool · Tablespace · Daemons & Background Processes

Oracle Architecture — SGA & Background Processes

System Global Area (SGA) is Oracle's shared memory structure

Shared Pool

Library cache + data dict. Stores parsed SQL, execution plans.

Database Buffer Cache

Cached copies of data blocks read from disk.

Redo Log Buffer

Records all changes before writing to redo log files.

Large Pool

Parallel query, RMAN backup, shared server memory.

Java Pool

Memory for Java-stored procedures and Java Virtual Machine.

Streams Pool

Used by Oracle Streams for data capture/apply.

Database Daemons & Background Processes

Always running silently — keeping your database alive and consistent

DBWn (DB Writer)

Writes dirty buffers from buffer cache to datafiles on disk.

LGWR (Log Writer)

Writes redo log buffer to online redo log files.

CKPT (Checkpoint)

Updates control file & datafile headers at checkpoint events.

SMON (Sys Monitor)

Instance recovery on startup; cleans temp segments.

PMON (Proc Monitor)

Cleans up failed user processes, releases locks/resources.

ARCn (Archiver)

Copies full redo logs to archive storage for recovery.

mysqld (MySQL)

Main MySQL daemon serving all client connections on port 3306.

postgres (PG)

PostgreSQL main process; spawns child for each connection.

sqlservr (MSSQL)

SQL Server engine process — listens on TCP 1433.

Tablespace, Buffer Pool & System Files

The physical storage layer of your database

Tablespace

A logical storage unit in Oracle/PostgreSQL. Contains segments→extents→blocks.

- SYSTEM tablespace: data dictionary
- USERS tablespace: user data
- TEMP tablespace: sort operations
- UNDO tablespace: rollback data

SQL Buffer Pool (MySQL InnoDB)

InnoDB Buffer Pool = RAM cache for data+index pages.

- Default: 128MB; production: 70-80% of RAM
- Key tuning: innodb_buffer_pool_size

Important DB System Files

Oracle: control files, datafiles (.dbf), redo logs, spfile, alert log

MySQL: ibdata1 (system tablespace), ib_logfile0/1, .frm, .ibd, my.cnf, error.log

PostgreSQL: pg_hba.conf, postgresql.conf, WAL segments in pg_wal/, base/ directory

MS SQL: .mdf (primary), .ndf (secondary), .ldf (log), SQL Server Error Log

All DBs: Audit logs, slow query logs, binary/transaction logs

SECTION 05

Database Keys

PK · FK · SK · AK · CK · Surrogate Key — Every key explained

All Database Keys Explained

Keys are the backbone of data integrity and relationships

PK

Primary Key

Uniquely identifies each row. Cannot be NULL. One per table.
Ex: employee_id INT PRIMARY KEY

FK/SK

Foreign Key (Secondary Key FK/SK)

References PK of another table. Enforces referential integrity.
Ex: dept_id INT REFERENCES departments(dept_id)

SK

Surrogate Key

Artificial system-generated key (usually AUTO_INCREMENT / SERIAL). Has no business meaning.
Ex: id SERIAL PRIMARY KEY

AK

Alternate Key

A candidate key NOT chosen as PK. Still unique. Also called Unique Key.
Ex: UNIQUE(email)

CK

Candidate Key

Any column/combo that COULD be PK. Unique + Not Null. One becomes PK, others become AK.
Ex: SSN or email could both be PK candidates

CK*

Composite Key

PK made of two or more columns combined.
Ex: PRIMARY KEY(order_id, product_id) in order_items table

SECTION 06

Tables, Records, Fields & Rows

The building blocks of relational data

Tables, Records, Fields & Rows

How data is physically organized in a relational database

employee_id (PK)	first_name	dept_id (FK)	salary
1001	Bob Miller	D01	$95,000
1002	John Smith	D02	$78,000
1003	Sara Ahmad	D01	$102,000

← TABLE: employees

← RECORD / ROW (one employee)

- TABLE = the container (like a spreadsheet tab)
- FIELD = a column (employee_id, first_name…)
- RECORD = one full row of data (one employee)
- ROW = same as record — a single data entry
- COLUMN = vertical — stores same data type
- CELL = intersection of row + column

SECTION 07

Normalization & Relationships

1NF · 2NF · 3NF — 1:1 · 1:M · M:N

Database Normalization — Why & How

Eliminate redundancy, prevent anomalies, ensure data integrity

1NF — First Normal Form

Rules:
• Each column holds atomic (indivisible) values
• No repeating groups or arrays
• Each row is unique

Example FIX:
phone: '555-1111, 555-2222'
→ Split into separate rows/table

2NF — Second Normal Form

Rules:
• Must be in 1NF
• Every non-key column fully depends on the WHOLE PK
(No partial dependency)

Example FIX:
order_items(order_id, prod_id, prod_name)
→ Move prod_name to products table

3NF — Third Normal Form

Rules:
• Must be in 2NF
• No transitive dependencies
(non-key col must depend only on PK)

Example FIX:
employee(emp_id, dept_id, dept_name)
→ dept_name depends on dept_id, not emp_id
→ Move to departments table

Database Normalization — Why & How (Boyce-Codd Normal Form (BCNF))

Eliminate redundancy, prevent anomalies, ensure data integrity

1NF — First Normal Form and (BCNF)

To understand normalization forms and Boyce-Codd Normal Form (BCNF), consider these key points:
Normalization Goal: Aim to reduce data redundancy and improve data integrity in relational databases.
First Normal Form (1NF): Ensure all columns contain atomic values and each record is unique.

2NF, 3NF Forms and (BCNF)

Second Normal Form (2NF): Achieve 1NF and eliminate partial dependencies on a composite primary key.
Third Normal Form (3NF): Meet 2NF and remove transitive dependencies, ensuring non-key attributes depend only on the primary key.

Boyce-Codd Normal Form (BCNF)

Boyce-Codd Normal Form (BCNF): Strengthen 3NF by ensuring every determinant is a candidate key, addressing anomalies in certain cases.
Normalization Process: Apply these forms progressively to refine database design and enhance query performance.

Database Relationships — 1:1, 1:M, M:N

How tables connect and relate to each other

1 : 1 (One-to-One)

One person has one passport.
Person —— Passport

Ex: employee ↔ employee_detail
Used for security / optional data.

1 : M (One-to-Many)

One customer has many orders.
Customer —< Orders

Most common relationship.
FK in the 'many' table points to PK in the 'one' table.

M : N (Many-to-Many)

Many students take many courses.
Student >—< Course

Requires a JUNCTION (bridge) table:

```
student_courses(
  student_id FK,
  course_id  FK,
  enrolled_date
)
```

Ex: Amazon orders ↔ products
Actors ↔ Movies

How to Normalize — Steps

- 1. Identify entities (nouns)
- 2. List all attributes per entity
- 3. Define PKs for each table
- 4. Apply 1NF: atomic values
- 5. Apply 2NF: remove partial deps
- 6. Apply 3NF: remove transitive deps
- 7. Define FK relationships
- 8. Draw ER Diagram (ERD)
- 9. Test with sample data

SECTION 08

DDL — Data Definition Language

CREATE · ALTER · DROP · TRUNCATE

DDL — Data Definition Language

Define and manage database structure — schema-level commands

```
-- Create a database
CREATE DATABASE university;
USE university;

-- Create a table
CREATE TABLE students (
    student_id  INT          PRIMARY KEY AUTO_INCREMENT,
    first_name  VARCHAR(50)  NOT NULL,
    last_name   VARCHAR(50)  NOT NULL,
    email       VARCHAR(100) UNIQUE,
    gpa         DECIMAL(3,2) DEFAULT 0.00,
    created_at  TIMESTAMP    DEFAULT CURRENT_TIMESTAMP
);

-- Create a user (MySQL)
CREATE USER 'webuser'@'localhost' IDENTIFIED BY 'SecurePass!';

-- Drop a table
DROP TABLE IF EXISTS old_students;

-- Truncate (delete all rows, keep structure)
TRUNCATE TABLE students;

-- Alter: add a column
ALTER TABLE students ADD COLUMN phone VARCHAR(20);
```

CREATE DATABASE

Creates a new empty database. In Oracle: CREATE TABLESPACE. PostgreSQL: CREATE DATABASE mydb;

DROP

Permanently removes database objects. DROP TABLE, DROP DATABASE, DROP INDEX. Cannot be rolled back!

ALTER / TRUNCATE

ALTER modifies existing structure. TRUNCATE removes all data fast but keeps schema.

SECTION 09

DML — Data Manipulation Language

INSERT · UPDATE · SELECT · DELETE

DML — Data Manipulation Language

Work with the actual data inside your tables

INSERT

```
-- INSERT
INSERT INTO students (first_name, last_name, email, gpa)
VALUES ('Wahidullah', 'Ahmadi', 'w@mwu.com', 3.95);

-- Bulk insert
INSERT INTO students (first_name, last_name, email)
VALUES ('John','Doe','j@x.com'),
       ('Sara','Lee','s@x.com');
```

SELECT

```
-- SELECT (basic to advanced)
SELECT * FROM students;
SELECT first_name, gpa FROM students WHERE gpa > 3.5;
SELECT s.first_name, c.course_name
FROM students s
JOIN enrollments e ON s.student_id = e.student_id
JOIN courses c     ON e.course_id  = c.course_id
WHERE s.gpa > 3.0
ORDER BY s.last_name ASC
LIMIT 10;
```

UPDATE

```
-- UPDATE
UPDATE students
SET gpa = 3.99, email = 'new@email.com'
WHERE student_id = 1001;
```

DELETE

```
-- DELETE (specific rows)
DELETE FROM students
WHERE student_id = 1001;
-- Use WHERE! No WHERE = deletes ALL rows.
```

SECTION 10

DCL — Data Control Language

GRANT · REVOKE — Security & Access Control

DCL — Data Control Language

Control who can access what — the security layer of SQL

GRANT

```
-- GRANT (give permissions)
-- MySQL
GRANT SELECT, INSERT, UPDATE
  ON university.students
  TO 'webuser'@'localhost';

-- Grant all on all tables
GRANT ALL PRIVILEGES
  ON university.*
  TO 'admin'@'%'
  WITH GRANT OPTION;

-- Oracle
GRANT SELECT ON students TO webuser;
GRANT CREATE SESSION TO webuser;

-- PostgreSQL
GRANT SELECT, INSERT ON students TO webuser;
```

REVOKE

```
-- REVOKE (remove permissions)
-- MySQL
REVOKE INSERT, UPDATE
  ON university.students
  FROM 'webuser'@'localhost';

-- Flush privileges (MySQL)
FLUSH PRIVILEGES;

-- Oracle
REVOKE SELECT ON students FROM webuser;

-- PostgreSQL
REVOKE ALL PRIVILEGES
  ON students FROM webuser;

-- Show grants (MySQL)
SHOW GRANTS FOR 'webuser'@'localhost';
```

SECTION 11

Distributed vs Centralized Databases

Replication · Snapshots · Redundancy · Banking Systems

Centralized vs Distributed Databases

Where your data lives — and why it matters

Centralized Database

All data in ONE location/server.

✅ Pros:
- Simple administration
- Easy backups
- No sync issues
- Consistent single source of truth

❌ Cons:
- Single point of failure
- Network latency for remote users
- Limited scalability

Example: Small business internal DB,
local university student system

Distributed Database

Data spread across MULTIPLE servers/locations.

✅ Pros:
- High availability (no single failure)
- Local data = low latency
- Horizontal scaling
- Geographic redundancy

❌ Cons:
- Complex sync & consistency
- CAP Theorem trade-offs
- Harder to administer

Examples: Bank (branches worldwide),
Amazon (multi-region), Google, Visa network

Replication, Snapshots & Redundancy

How databases ensure data survives failures

Replication

Copies data from master → slave(s) in real time or near-real time.

Types:
• Synchronous: all replicas updated before commit confirms
• Asynchronous: master commits, replicas catch up
• Multi-master: writes accepted anywhere

Used by: MySQL Replication, Oracle Data Guard, PostgreSQL Streaming

Snapshots

A point-in-time copy of the database state.

Types:
• Hot snapshot: taken while DB is live
• Cold snapshot: DB stopped first
• Logical: SQL dump (mysqldump)
• Physical: block-level copy

Used for: backups, reporting, testing, dev clones

Tools: RMAN (Oracle), mysqldump, pg_dump, LVM snapshot

Redundancy & HA

Multiple copies protect against data loss.

Techniques:
• RAID storage arrays
• Standby / failover nodes
• Active-Passive clustering
• Active-Active (load balanced)
• Geographic failover (DR site)

Banking Example:
Visa processes 65,000 TPS with zero tolerance for downtime — uses 99.999% uptime SLA across global distributed nodes.

Querying Flat Files vs, a Database

Why a real database beats Excel/CSV at scale

Feature	Excel / CSV / Flat File	Relational Database
Max Records	~1M rows in Excel	Billions of rows (no limit)
Concurrent Users	One user at a time	Thousands simultaneously
Data Integrity	No enforcement	PK, FK, constraints enforced
Querying	Manual filters, formulas	SQL — instant, indexed queries
Security	File permissions only	Row/column-level security
Transactions	No ACID support	Full ACID compliance
Relationships	VLOOKUP (fragile)	JOINs — fast & reliable
Backup/Recovery	Manual copy	Automated, point-in-time recovery
Scalability	Breaks at large size	Scales vertically & horizontally

SECTION 12

Data Science, APIs & Careers

Python · Java · C++ · ODBC · APIs · DBA Jobs

Connecting to Databases — Languages & APIs

Python, Java, C/C++, R, ODBC, RPC, REST API

Python (mysql-connector, psycopg2, pandas, SQLAlchemy)

```
import mysql.connector          # MySQL
import psycopg2                    # PostgreSQL
import pyodbc                      # ODBC (any DB)
import pandas as pd
import sqlalchemy as sa

# Connect
conn = mysql.connector.connect(
    host="localhost", user="root",
    password="pass", database="university")
cursor = conn.cursor()

# Query into DataFrame
df = pd.read_sql(
    "SELECT * FROM students WHERE gpa > 3.5",

con=sa.create_engine("mysql+mysqlconnector://root:pass@localhost/univ
ersity"))
print(df.head())
conn.close()
```

Java JDBC + C/C++ (libmysqlclient / ODBC)

```
// Java JDBC
import java.sql.*;
Connection conn = DriverManager.getConnection(
    "jdbc:mysql://localhost:3306/university",
    "root", "password");
PreparedStatement ps = conn.prepareStatement(
    "SELECT * FROM students WHERE gpa > ?");
ps.setDouble(1, 3.5);
ResultSet rs = ps.executeQuery();
while(rs.next()){
    System.out.println(rs.getString("first_name"));
}
```

ODBC · RPC · REST API — Connecting Applications to Databases

ODBC

Open Database Connectivity — universal driver interface.

- Works with any ODBC-compatible DB
- pyodbc (Python), JDBC-ODBC bridge (Java)
- Configure DSN in Windows ODBC Admin or /etc/odbc.ini

Example:
pyodbc.connect('DSN=MySQL;UID=root;PWD=pass')

Used heavily in: Excel → DB, Power BI, Tableau

RPC & gRPC

Remote Procedure Call — call functions on remote DB servers.

- Traditional RPC: older, platform-specific
- gRPC: Google's modern high-performance RPC using Protocol Buffers
- Used for microservices talking to databases

Flow:
Client → gRPC stub → Network → gRPC server → DB query → response

Used by: Google, Netflix, Uber backends

REST API

Most common way web apps talk to databases.

Flow:
Frontend → HTTP REST API → Backend (Node/Python/Java) → SQL query → DB

HTTP Methods → SQL:
GET → SELECT
POST → INSERT
PUT → UPDATE
DELETE → DELETE

Tools: FastAPI, Django REST, Spring Boot, Express.js
Auth: JWT tokens, OAuth2, API keys

Data Science with Databases & Datasets

SQL is the #1 skill for data scientists — more than Python!

Key Data Science DB Tools

- pandas + SQLAlchemy → query DB directly into DataFrames
- Jupyter Notebook → interactive SQL + visualizations
- Apache Spark → SQL over huge distributed datasets
- Tableau / Power BI → drag-drop SQL-powered dashboards
- dbt (data build tool) → SQL-based data transformations
- R language → DBI + dplyr + dbplyr for DB queries

Public Datasets to Practice With

- Kaggle.com — thousands of CSV/SQL datasets
- data.gov — US government open data
- UCI ML Repository — classic ML datasets
- Google BigQuery Public Datasets (free tier)
- AWS Open Data Registry

R Language + Database (DBI, RMySQL, dplyr)

```r
library(DBI); library(RMySQL)
con <- dbConnect(RMySQL::MySQL(),
  dbname="university", host="localhost",
  user="root", password="pass")

df <- dbGetQuery(con,
  "SELECT * FROM students
   WHERE gpa > 3.5
   ORDER BY gpa DESC")

# dplyr lazy evaluation
library(dplyr)
students_tbl <- tbl(con, "students")
students_tbl %>%
  filter(gpa > 3.5) %>%
  arrange(desc(gpa)) %>%
  collect()   # Execute SQL query

dbDisconnect(con)
```

🚀 DBA Career Guide — Entry to Senior Level

Database Administrator is one of the highest-paid tech roles

Role	Skills Required	Avg Salary (US)	Where to Apply
Junior DBA	SQL basics, backup/restore, Linux, 1 DBMS	$65–85K	LinkedIn, Indeed, Glassdoor, USAJobs.gov
Mid-Level DBA	Performance tuning, replication, scripting	$85–115K	Dice.com, Hired.com, company career pages
Senior Oracle DBA	RAC, DataGuard, RMAN, OCP certification	$120–160K	Oracle, banks, hospitals, airlines
Data Engineer	SQL + Python + Spark + ETL + cloud	$110–145K	Airbnb, Netflix, Amazon, Uber, Meta
Data Scientist	SQL + Python/R + ML + statistics + visualization	$115–165K	Google, Apple, startups, finance, pharma
Cloud DBA (AWS/Azure)	AWS RDS/Aurora, Azure SQL, Terraform, IaC	$130–175K	AWS, Microsoft, Accenture, Deloitte

📜 Certifications, Tools & Learning Resources

Your roadmap to becoming a Database Professional

Top Certifications

Oracle:
- OCA (Associate)
- OCP (Professional)
- OCM (Master)

MySQL:
- MySQL Developer
- MySQL DBA

Microsoft:
- DP-300 (Azure SQL)
- 70-764 (SQL Server)

PostgreSQL:
- EDB Postgres Professional

Cloud:
- AWS Database Specialty
- Google Cloud Professional Data Engineer

Essential Tools

GUI Clients:
- MySQL Workbench (free)
- DBeaver (free, all DBs)
- pgAdmin (PostgreSQL)
- SQL Server Management Studio
- Oracle SQL Developer
- TablePlus

Command Line:
mysql, psql, sqlplus, sqlcmd

Monitoring:
- Oracle Enterprise Manager
- Percona Monitoring
- pgBadger
- Prometheus + Grafana

Free Learning Resources

Websites:
- MyWebUniversity.com ⭐
- dev.mysql.com/doc
- postgresql.org/docs
- docs.oracle.com
- w3schools.com/sql
- sqlzoo.net
- leetcode.com (SQL section)

YouTube:
- @MyWebUniversity ⭐
- @MyWebUniversity-Dari ⭐

Practice Platforms:
- SQLFiddle.com
- db-fiddle.com
- Mode Analytics
- Kaggle SQL courses

Thank You!

You are now ready to start your SQL & Database journey!

MyWebUniversity.com

youtube.com/@MyWebUniversity

youtube.com/@MyWebUniversity-Dari

Support: ko-fi.com/mywebuniversity

Free Linux, Unix, Windows & Programming Tutorials

Questions? Comments? Leave them below!

LIKE SUBSCRIBE SHARE COMMENT