**DBMS LAB PROJECT** **MOVENTUR:**

**NORMALIZED RELATIONAL**

**SCHEMA**

**KEY MILESTONE # 0****2**

**Spring 2025**

**CSE-403L**

**Database Management System Lab**

Submitted by:

**Muhammad Musa (22PWCSE2157)**

**Muhammad Jasim (22PWCSE2120)**

Class Section: **‘C’**

“We affirm that we have completed this work with integrity”

Submitted to:

**Engr. Sumayyea Salahuddin**

May 28, 2025

Department of Computer Systems Engineering

University of Engineering and Technology, Peshawar

**Project Normalized Relational Schema**

This document outlines the conversion of the provided Entity-Relationship Diagram (ERD) into a relational schema and its subsequent normalization to Third Normal Form (3NF).

**1. Converted Relational Schema (from ERD)**

Based on the provided ERD, here is the initial relational schema:

* **CANDIDATES** (<u>id</u>, position\_id, firstname, lastname, photo, platform)
* **POSITIONS** (<u>id</u>, description, max\_vote, priority)
* **ADMIN** (<u>id</u>, username, password, voters, votes)
* **VOTERS** (<u>id</u>, voters\_id, password, firstname, lastname, photo)
* **VOTES** (<u>id</u>, voter\_id, candidate\_id, position\_id)

**Note on Relationships:**

* A one-to-many relationship from POSITIONS to CANDIDATES is represented by position\_id in CANDIDATES as a foreign key referencing POSITIONS.
* The relationships involving ADMIN with VOTERS and VOTES (implied by voters and votes attributes in ADMIN) need careful consideration during normalization, as these attributes likely represent counts or indirect associations rather than direct foreign keys. For a true relational schema, these would typically be derived or managed through relationships with other tables. Assuming voters and votes in ADMIN are descriptive attributes for the admin's activity rather than direct foreign keys, they will be handled in normalization.
* The many-to-many relationship between VOTERS and CANDIDATES (via VOTES) is represented by the VOTES table, which acts as an associative entity.

**2. Step-wise Normalization to 3NF**

We will normalize each relation step-by-step to achieve Third Normal Form (3NF).

**Relation: CANDIDATES**

**Initial Schema:** CANDIDATES (<u>id</u>, position\_id, firstname, lastname, photo, platform)

**Primary Key:** id

**Functional Dependencies (FDs):**

* id rightarrow position\_id, firstname, lastname, photo, platform (All attributes are dependent on the primary key)

**1NF (First Normal Form):**

The relation is already in 1NF as all attributes are atomic (single-valued) and there are no repeating groups.

**2NF (Second Normal Form):**

The relation is in 2NF because it is in 1NF and all non-key attributes (position\_id, firstname, lastname, photo, platform) are fully functionally dependent on the primary key (id). There are no partial dependencies since the primary key is a single attribute.

**3NF (Third Normal Form):**

The relation is in 3NF because it is in 2NF and there are no transitive dependencies. No non-key attribute is dependent on another non-key attribute.

**Normalized Relation:**

* **CANDIDATES** (<u>id</u>, position\_id, firstname, lastname, photo, platform)

**Relation: POSITIONS**

**Initial Schema:** POSITIONS (<u>id</u>, description, max\_vote, priority)

**Primary Key:** id

**Functional Dependencies (FDs):**

* id rightarrow description, max\_vote, priority (All attributes are dependent on the primary key)

**1NF (First Normal Form):**

The relation is already in 1NF as all attributes are atomic and there are no repeating groups.

**2NF (Second Normal Form):**

The relation is in 2NF because it is in 1NF and all non-key attributes (description, max\_vote, priority) are fully functionally dependent on the primary key (id). There are no partial dependencies.

**3NF (Third Normal Form):**

The relation is in 3NF because it is in 2NF and there are no transitive dependencies.

**Normalized Relation:**

* **POSITIONS** (<u>id</u>, description, max\_vote, priority)

**Relation: ADMIN**

**Initial Schema:** ADMIN (<u>id</u>, username, password, voters, votes)

**Primary Key:** id

**Functional Dependencies (FDs):**

* id rightarrow username, password, voters, votes

**Consideration for voters and votes attributes:** Based on common database design practices, attributes like voters and votes in an ADMIN table are highly likely to be *derived* attributes (e.g., counts of voters managed or votes processed by an admin) rather than direct data that belongs in the ADMIN table itself. Storing derived data can lead to update anomalies if not carefully managed. For strict normalization, derived attributes are often removed.

Assuming voters and votes are *not* direct foreign keys or essential, non-derived properties of the Admin entity itself, they should be removed or calculated via queries. If they are meant to be direct counts, they violate 3NF if they depend on something other than the primary key or are derivable from other tables. For the purpose of normalization, we will assume they are *not* directly part of the ADMIN entity's core data and would be better handled as derived values or through separate audit/log tables if they represent actions.

**Revised Initial Schema (removing potentially derived attributes for normalization):** ADMIN (<u>id</u>, username, password)

**Primary Key:** id

**Functional Dependencies (FDs):**

* id rightarrow username, password
* username rightarrow password (assuming username is unique and determines password)

**1NF (First Normal Form):**

The relation is already in 1NF.

**2NF (Second Normal Form):**

The relation is in 2NF.

**3NF (Third Normal Form):**

The relation is in 3NF if username is a candidate key. If username is unique and determines password, then username is a candidate key. If id is the *only* primary key and username is just a unique attribute, then password is transitively dependent on id through username.

To ensure 3NF, if username is not the primary key, and password depends on username, we would decompose:

**Decomposition for 3NF (if username is a determinant for password and not the primary key):**

* **ADMIN** (<u>id</u>, username)
* **ADMIN\_CREDENTIALS** (<u>username</u>, password)

However, typically in an ADMIN table, username is often considered unique and part of the identifying information. If username is a candidate key, then the original ADMIN (<u>id</u>, username, password) is already in 3NF provided password is fully dependent on username (which it is).

Given the common practice for user tables, let's assume username is unique and password directly relates to the username. The id is the surrogate primary key.

**Normalized Relation (assuming username is unique and password depends on it, but id is the primary key):**

* **ADMIN** (<u>id</u>, username, password)
  + *Note:* If username is guaranteed unique, it could also be a candidate key. The current structure is acceptable in 3NF if password is considered directly dependent on the id (via the unique username associated with that id).

**Relation: VOTERS**

**Initial Schema:** VOTERS (<u>id</u>, voters\_id, password, firstname, lastname, photo)

**Primary Key:** id

**Functional Dependencies (FDs):**

* id rightarrow voters\_id, password, firstname, lastname, photo
* voters\_id rightarrow password, firstname, lastname, photo (Assuming voters\_id is a unique identifier for a voter, similar to a national ID or unique registration number, and determines other attributes).

**1NF (First Normal Form):**

The relation is already in 1NF.

**2NF (Second Normal Form):**

The relation is in 2NF.

**3NF (Third Normal Form):**

If voters\_id is a candidate key (unique and determines other attributes), and id is the primary key, then there is a transitive dependency: id rightarrow voters\_id rightarrow password, firstname, lastname, photo.

To remove this transitive dependency, we decompose the table:

**Decomposition for 3NF:**

* **VOTERS** (<u>id</u>, voters\_id)
* **VOTER\_DETAILS** (<u>voters\_id</u>, password, firstname, lastname, photo)

**Normalized Relations:**

* **VOTERS** (<u>id</u>, voters\_id)
* **VOTER\_DETAILS** (<u>voters\_id</u>, password, firstname, lastname, photo)

**Relation: VOTES**

**Initial Schema:** VOTES (<u>id</u>, voter\_id, candidate\_id, position\_id)

**Primary Key:** id (This id is likely a surrogate key for the vote record itself)

**Candidate Keys:**

* (voter\_id, position\_id) - A voter typically casts one vote for a specific position. This combination should uniquely identify a vote.
* (id) - The surrogate primary key.

**Functional Dependencies (FDs):**

* id rightarrow voter\_id, candidate\_id, position\_id
* (voter\_id, position\_id) rightarrow candidate\_id (A voter for a specific position will vote for one candidate for that position).

**1NF (First Normal Form):**

The relation is already in 1NF.

**2NF (Second Normal Form):**

The relation is in 2NF.

**3NF (Third Normal Form):**

The relation is in 3NF because there are no transitive dependencies. candidate\_id is fully dependent on the composite candidate key (voter\_id, position\_id).

**Normalized Relation:**

* **VOTES** (<u>id</u>, voter\_id, candidate\_id, position\_id)

**Final Normalized Relational Schema (3NF)**

Here is the complete set of relations in Third Normal Form (3NF) for the Online Voting System:

* **CANDIDATES** (<u>id</u>, position\_id, firstname, lastname, photo, platform)
  + position\_id is a Foreign Key (FK) referencing POSITIONS(id).
* **POSITIONS** (<u>id</u>, description, max\_vote, priority)
* **ADMIN** (<u>id</u>, username, password)
  + *Note:* username is assumed to be unique.
* **VOTERS** (<u>id</u>, voters\_id)
  + voters\_id is a Foreign Key (FK) referencing VOTER\_DETAILS(voters\_id).
* **VOTER\_DETAILS** (<u>voters\_id</u>, password, firstname, lastname, photo)
  + voters\_id is the Primary Key (PK).
* **VOTES** (<u>id</u>, voter\_id, candidate\_id, position\_id)
  + voter\_id is a Foreign Key (FK) referencing VOTERS(id).
  + candidate\_id is a Foreign Key (FK) referencing CANDIDATES(id).
  + position\_id is a Foreign Key (FK) referencing POSITIONS(id).
  + The composite key (voter\_id, position\_id) ensures a voter can only vote once per position.