Normalization in Project

Relation	Primary Key	Functional Dependencies (FDs)	2NF Check (Partial Dependencies)	3NF Check (Transitive Dependencies)	Conclusion
Members	member_id	member_id → fname, lname, email, password, birthday, gender, class_count email → member_id, ...	PK is single attribute → No partial dependencies	No non-key determines another non-key → No transitive dependencies	In 3NF
Metrics	metric_id	metric_id → member_id, metric_date, weight, body_fat, heart_rate	Single PK → No partial dependencies	All non-key attributes depend only on key → No transitive dependencies	In 3NF
Goals	goal_id	goal_id → member_id, metric_name, current_metric, goal_metric (member_id, metric_name) → current_metric, goal_metric	Single PK → No partial dependencies	No non-key → non-key dependencies → No transitive dependencies	In 3NF
Trainers	trainer_id	trainer_id → fname, lname, email, password, specialization	Single PK → No partial dependencies	No non-key → non-key dependencies → No transitive dependencies	In 3NF
Classes	class_id	class_id → booking_id, trainer_id, attendance booking_id → class_id, trainer_id, attendance	Single PK → No partial dependencies	No non-key → non-key dependencies → No transitive dependencies	In 3NF
Class_Regs	reg_id	reg_id → class_id, member_id	Single PK → No partial dependencies	No non-key → non-key dependencies → No transitive dependencies	In 3NF

***CONCLUSION

Based on the analysis above, all functions in the DB satisfy 2NF because ***no relation has a composite primary key,*** and **THEREFORE** partial dependency cannot occur. Also all relations satisfy 3NF because ***no non-key attribute depends on another non-key attribute***.
Any stored derived values (class_count, attendance, current_metric) are maintained through triggers or application logic and therefore do not violate normal form requirements.**Thus, the schema is fully normalized to 3NF.**