Inhalt

[Reading 3](#_Toc447558299)

[neo4j\_getUserByID 3](#_Toc447558300)

[neo4j\_getUserByEmail 3](#_Toc447558301)

[neo4j\_authUserByEmail 3](#_Toc447558302)

[neo4j\_getUserByFB 3](#_Toc447558303)

[neo4j\_getWorkoutByID 3](#_Toc447558304)

[neo4j\_getExerciseByID 3](#_Toc447558305)

[neo4j\_getRelevantTags 3](#_Toc447558306)

[neo4j\_searchTag 4](#_Toc447558307)

[neo4j\_getRandomWorkouts 4](#_Toc447558308)

[neo4j\_getMostLikedWorkouts 4](#_Toc447558309)

[neo4j\_getTaggedWorkouts 4](#_Toc447558310)

[neo4j\_getCustomizedWorkouts 4](#_Toc447558311)

[neo4j\_getPersonalizedWorkouts 5](#_Toc447558312)

[neo4j\_getNearWorkoutWorkouts 5](#_Toc447558313)

[neo4j\_getOrderedTaggedWorkouts( 5](#_Toc447558314)

[neo4j\_getUserFavs 5](#_Toc447558315)

[neo4j\_getUser**Create**dWorkouts 5](#_Toc447558316)

[neo4j\_searchWorkout 6](#_Toc447558317)

[neo4j\_getWorkoutExercises 6](#_Toc447558318)

[neo4j\_getWorkoutTags 6](#_Toc447558319)

[neo4j\_getRandomExercises 7](#_Toc447558320)

[neo4j\_getSimilarExercises 7](#_Toc447558321)

[neo4j\_searchExercise 7](#_Toc447558322)

[neo4j\_searchUserByName 7](#_Toc447558323)

[neo4j\_getFollowers 8](#_Toc447558324)

[neo4j\_userFollowedUser 8](#_Toc447558325)

[neo4j\_userLikedWorkout 8](#_Toc447558326)

[neo4j\_getPWRe**set**Token 8](#_Toc447558327)

[Writing 8](#_Toc447558328)

[neo4j\_**create**Tag 8](#_Toc447558329)

[neo4j\_workoutHasTag 8](#_Toc447558330)

[neo4j\_removeWorkoutHasTag 9](#_Toc447558331)

[neo4j\_exerciseHasTag 9](#_Toc447558332)

[neo4j\_removeExerciseHasTag 9](#_Toc447558333)

[neo4j\_**create**Workout 9](#_Toc447558334)

[neo4j\_workoutHasExercise 9](#_Toc447558335)

[neo4j\_removeWorkoutHasExercises 9](#_Toc447558336)

[neo4j\_updateWorkout 10](#_Toc447558337)

[neo4j\_removeWorkout 10](#_Toc447558338)

[neo4j\_likeDislikeWorkout 10](#_Toc447558339)

[neo4j\_**create**User 10](#_Toc447558340)

[neo4j\_get3doUser 10](#_Toc447558341)

[neo4j\_markRememberedUser 10](#_Toc447558342)

[neo4j\_confirmUserMail 11](#_Toc447558343)

[neo4j\_followUnfollow 11](#_Toc447558344)

[neo4j\_updateUserProfile 11](#_Toc447558345)

[neo4j\_**set**PWRe**set**Token 11](#_Toc447558346)

[neo4j\_**set**UserPassword 11](#_Toc447558347)

[neo4j\_removeUser 11](#_Toc447558348)

[neo4j\_**create**Exercise 12](#_Toc447558349)

[neo4j\_exerciseSimilar 12](#_Toc447558350)

[neo4j\_removeExerciseSimilar 12](#_Toc447558351)

[neo4j\_updateExercise 12](#_Toc447558352)

[Metrics 12](#_Toc447558353)

[neo4j\_basicMetrics 12](#_Toc447558354)

# Reading

## neo4j\_getUserByID

**MATCH** (u:User {user\_id: {user\_id} })

**RETURN** u.name, u.user\_id, u.email, u.confirmed, u.contact, u.visible, u.shortBio, u.locale, u.picPath **LIMIT** 1

## neo4j\_getUserByEmail

**MATCH** (u:User {email: {email} })

**RETURN** u.name, u.user\_id, u.confirmed, u.contact, u.visible, u.shortBio, u.locale, u.picPath **LIMIT** 1

## neo4j\_authUserByEmail

**MATCH** (u:User {email: {email} })

**RETURN** u.name, u.email, u.user\_id, u.confirmed, u.password, u.contact, u.visible, u.shortBio, u.locale, u.picPath **LIMIT** 1

## neo4j\_getUserByFB

**MATCH** (u:User {fbid: {fbid} })

**RETURN** u.name, u.email, u.user\_id, u.confirmed, u.password, u.contact, u.visible, u.shortBio, u.locale, u.picPath **LIMIT** 1

## neo4j\_getWorkoutByID

**MATCH** (w:Workout {workout\_id: {workout\_id} })

**WITH** w **MATCH** (w)<-[:**CREATE**D]-(u:User)

**RETURN** w.name, w.workout\_id, w.dimensions, w.difficulty, w.duration, w.roundCount, w.exCount, w.picPath, u.name, u.user\_id, u.picPath, u.shortBio **LIMIT** 1

## neo4j\_getExerciseByID

**MATCH** (e:Exercise {exercise\_id: {exercise\_id} })

**RETURN** e.name, e.exercise\_id, e.alias, e.duration, e.picPath, e.shortDescr, e.execModi, e.dimensions, e.equipment, e.difficulty **LIMIT** 1

## neo4j\_getRelevantTags

**MATCH** (t:Tag)<-[r:HAS\_TAG

**RETURN** t.name, count(r) as connections

**ORDER BY** connections DESC **LIMIT** {maximum}

## neo4j\_searchTag

**MATCH** (t:Tag)

**WHERE** t.name =~ {searchWord}

**RETURN** t.name **ORDER BY** t.name ASC **LIMIT** {maximum}

## neo4j\_getRandomWorkouts

**MATCH** (w:Workout)

**WITH** w, rand() as rand **ORDER BY** rand **LIMIT** 1000

**MATCH** (w)<-[:**CREATE**D]-(u:User)

**RETURN** w.name, w.workout\_id, w.dimensions, w.duration, w.difficulty, w.roundCount, w.exCount, w.picPath, u.name, u.user\_id, u.picPath, u.shortBio

**SKIP** {pagination} **LIMIT** {maximum}

## neo4j\_getMostLikedWorkouts

**MATCH** (w:Workout)<-[r:LIKED]-()

**WITH** w **MATCH** (w)-[:HAS\_TAG]->(t:Tag) **WHERE** t.name IN {tags}

**WITH** w, count(r) as likes **ORDER BY** likes DESC **SKIP** {pagination} **LIMIT** {maximum}

**MATCH** (w)<-[:**CREATE**D]-(u:User)

**RETURN** w.name, w.workout\_id, w.dimensions, w.duration, w.difficulty, w.roundCount, w.exCount, w.picPath, u.name, u.user\_id, u.picPath, u.shortBio

**SKIP** {pagination} **LIMIT** {maximum}

## neo4j\_getTaggedWorkouts

**MATCH** (t:Tag)<-[r:HAS\_TAG]-(w:Workout)

**WHERE** t.name IN {tags} AND w.difficulty >= {difficultyMin} AND w.difficulty <= {difficultyMax} AND w.duration >= {timeMin} AND w.duration <= {timeMax}

**WITH** w, count(r) as tag**Match**es **ORDER BY** tag**Match**es DESC **SKIP** {pagination} **LIMIT** {maximum}

**MATCH** (w)<-[:**CREATE**D]-(u:User)

**RETURN** tag**Match**es, w.name, w.workout\_id, w.dimensions, w.duration, w.difficulty, w.roundCount, w.exCount, w.picPath, u.name, u.user\_id, u.picPath, u.shortBio

**ORDER BY** tag**Match**es DESC **SKIP** {pagination} **LIMIT** {maximum}

## neo4j\_getCustomizedWorkouts

MATCH (t:Tag {name: "Fun with Friends"}) WITH t   
MATCH (t)<-[:HAS\_TAG]-(w:Workout)   
WHERE w.difficulty >= {diffMin} AND w.difficulty <= {diffMax}   
WITH COLLECT(w) AS resultSet1 MATCH (t:Tag {name: "Be Fit"}) WITH t , resultSet1   
MATCH (t)<-[:HAS\_TAG]-(w:Workout)   
WHERE w.difficulty >= {diffMin} AND w.difficulty <= {diffMax}   
WITH COLLECT(w) AS resultSet2, resultSet1   
MATCH (t:Tag {name: "Inspirational"}) WITH t , resultSet1, resultSet2   
MATCH (t)<-[:HAS\_TAG]-(w1:Workout)   
WHERE w1.difficulty >= {diffMin} AND w1.difficulty <= {diffMax}   
WITH COLLECT(w1) as col1 , resultSet1, resultSet2   
MATCH (w2:Workout)   
WHERE w2.exCount > 6 AND w2.difficulty >= "1.0" AND w2.difficulty <= "5.0"   
WITH COLLECT(w2) + col1 AS all , resultSet1, resultSet2   
UNWIND all AS w WITH DISTINCT w , resultSet1, resultSet2   
WITH COLLECT(w) AS resultSet3, resultSet1, resultSet2   
MATCH (w1:Workout)-[r1:HAS\_EXERCISE]->(:Exercise)   
WHERE w1.duration <= 600 AND w1.difficulty >= "3.0" AND w1.difficulty <= "5.0"   
WITH w1, count(r1) AS totalExercises , resultSet1, resultSet2, resultSet3 MATCH (w1)-[r2:HAS\_EXERCISE]->(e2:Exercise) WHERE e2.difficulty >= "4" WITH w1, totalExercises, count(r2) AS diffExercises , resultSet1, resultSet2, resultSet3 WHERE TOFLOAT(diffExercises)/totalExercises >= 0.3 WITH COLLECT(w1) AS col1 , resultSet1, resultSet2, resultSet3 MATCH (t1:Tag)<-[:HAS\_TAG]-(w2:Workout) WHERE t1.name IN ["Beast Mode", "EMOM", "Tabata"] AND w2.duration <= 600 AND w2.difficulty >= "3.0" AND w2.difficulty <= "5.0" WITH COLLECT(w2) + col1 AS all , resultSet1, resultSet2, resultSet3 UNWIND all AS w WITH DISTINCT w , resultSet1, resultSet2, resultSet3 WITH COLLECT(w) AS resultSet4, resultSet1, resultSet2, resultSet3 MATCH (w1:Workout)-[r1:HAS\_EXERCISE]->(:Exercise) WHERE w1.difficulty >= {diffMin} AND w1.difficulty <= {diffMax} WITH w1, count(r1) AS totalExercises , resultSet1, resultSet2, resultSet3, resultSet4 MATCH (w1)-[r2:HAS\_EXERCISE]->(e2:Exercise) WHERE e2.difficulty >= "4" WITH w1, totalExercises, count(r2) AS diffExercises , resultSet1, resultSet2, resultSet3, resultSet4 WHERE TOFLOAT(diffExercises)/totalExercises >= 0.5 WITH COLLECT(w1) AS col1 , resultSet1, resultSet2, resultSet3, resultSet4 MATCH (t)<-[:HAS\_TAG]-(w2:Workout) WHERE w2.duration >= 25 \* 60 AND w2.difficulty >= "1.0" AND w2.difficulty <= "5.0" WITH COLLECT(w2) AS col2, col1 , resultSet1, resultSet2, resultSet3, resultSet4 MATCH (t:Tag)<-[:HAS\_TAG]-(w3:Workout) WHERE t.name = "Beast Mode" AND w3.difficulty >= "1.0" AND w3.difficulty <= "5.0" WITH COLLECT(w3) + col1 + col2 AS all , resultSet1, resultSet2, resultSet3, resultSet4 UNWIND all AS w WITH DISTINCT w , resultSet1, resultSet2, resultSet3, resultSet4 WITH COLLECT(w) AS resultSet5, resultSet1, resultSet2, resultSet3, resultSet4 MATCH (w1:Workout)-[:HAS\_EXERCISE]->(e1:Exercise) WHERE e1.equipment = false AND w1.difficulty >= {diffMin} AND w1.difficulty <= {diffMax} WITH COLLECT(w1) AS col1 , resultSet1, resultSet2, resultSet3, resultSet4, resultSet5 OPTIONAL MATCH (t:Tag {name: "Travel"})<-[:HAS\_TAG]-(w2:Workout) WHERE w2.difficulty >= {diffMin} AND w2.difficulty <= {diffMax} WITH COLLECT(w2) + col1 AS all , resultSet1, resultSet2, resultSet3, resultSet4, resultSet5 UNWIND all AS w WITH DISTINCT w , resultSet1, resultSet2, resultSet3, resultSet4, resultSet5 WITH COLLECT(w) AS resultSet6, resultSet1, resultSet2, resultSet3, resultSet4, resultSet5 WITH resultSet1 + resultSet2 + resultSet3 + resultSet4 + resultSet5 + resultSet6 AS all UNWIND all AS w WITH DISTINCT w RETURN w, (CASE WHEN w.dimensions[0] >= 0.5 THEN w.dimensions[0] ELSE 0.0 END + CASE WHEN w.dimensions[1] >= 0.5 THEN w.dimensions[1] ELSE 0.0 END + CASE WHEN w.dimensions[2] >= 0.5 THEN w.dimensions[2] ELSE 0.0 END + CASE WHEN w.dimensions[3] >= 0.5 THEN w.dimensions[3] ELSE 0.0 END) AS dimMatch ORDER BY dimMatch DESC SKIP 0 LIMIT 10

## neo4j\_getPersonalizedWorkouts

**MATCH** (u:User {user\_id: {user\_id} })-[:LIKED]->(wu:Workout)-[r1:HAS\_TAG]->(t:Tag)

**WITH** u, t, count(r1) as commonTags

**MATCH** (t)<-[:HAS\_TAG]-(w:Workout)<-[r2]-(u)

**WHERE** r2 IS NULL

**WITH** w, commonTags **MATCH** (w)<-[:**CREATE**D]-(u:User)

**RETURN** w.name, w.workout\_id, w.dimensions, w.duration, w.difficulty, w.roundCount, w.exCount, w.picPath, u.name, u.user\_id, u.picPath, u.shortBio

**ORDER BY** commonTags DESC **SKIP** {pagination} **LIMIT** {maximum}

## neo4j\_getNearWorkoutWorkouts

**MATCH** (w1:Workout {workout\_id: {workout\_id} })-[:HAS\_TAG]->(t:Tag)<-[:HAS\_TAG]-(w:Workout)

**WITH** w, count(\*) as commonTags **MATCH** (w)<-[:**CREATE**D]-(u:User )

**RETURN** w.name, w.workout\_id, w.dimensions, w.duration, w.difficulty, w.roundCount, w.exCount, w.picPath, u.name, u.user\_id, u.picPath, u.shortBio

**ORDER BY** commonTags **SKIP** {pagination} **LIMIT** {maximum}

## neo4j\_getOrderedTaggedWorkouts(

**MATCH** (w:Workout)-[:HAS\_TAG]->(t:Tag)

**WHERE** t.name IN {tags}

**WITH** w **MATCH** (w)<-[:**CREATE**D]-(u:User)

**RETURN** w.name, w.workout\_id, w.dimensions, w.duration, w.picPath, w.{propName} as orderProp, w.difficulty, w.roundCount, w.exCount, w.picPath, u.name, u.user\_id, u.picPath, u.shortBio

**ORDER BY** orderProp DESC **SKIP** {pagination} **LIMIT** {maximum}

## neo4j\_getUserFavs

**MATCH** (u2:User {user\_id: {user\_id} })-[:LIKED]->(w:Workout)<-[:**CREATE**D]-(u:User)

**RETURN** w.name, w.workout\_id, w.dimensions, w.duration, w.difficulty, w.roundCount, w.exCount, w.picPath, u.name, u.user\_id, u.picPath, u.shortBio

**SKIP** {**skip**} **LIMIT** {maximum}

## neo4j\_getUser**Create**dWorkouts

**MATCH** (u:User {user\_id: {user\_id} })-[:**CREATE**D]->(w:Workout)

**RETURN** w.name, w.workout\_id, w.dimensions, w.duration, w.difficulty, w.roundCount, w.exCount, w.picPath, u.name, u.user\_id, u.picPath, u.shortBio

**SKIP** {**skip**} **LIMIT** {maximum}

## neo4j\_searchWorkout

**MATCH** (w:Workout)

**WHERE** w.name =~ {searchWordRegex} AND w.difficulty >= {difficultyMin} AND w.difficulty <= {difficultyMax} AND w.duration >= {timeMin} AND w.duration <= {timeMax}

**WITH** w **MATCH** (w)<-[:**CREATE**D]-(u:User)

**RETURN** w.name, ABS( LENGTH(w.name) - LENGTH({searchWord}) ) AS nameDiff, w.workout\_id, w.dimensions, w.duration, w.difficulty, w.roundCount, w.exCount, w.picPath, u.name, u.user\_id, u.picPath, u.shortBio

**ORDER BY** nameDiff, w.name ASC **SKIP** {**skip**} **LIMIT** {maximum}

**UNION**

**MATCH** (u:User)-[:**CREATE**D]->(w:Workout)-[r:HAS\_TAG]->(t:Tag)

**WHERE** t.name =~ {searchWordRegex} AND w.difficulty >= {difficultyMin} AND w.difficulty <= {difficultyMax} AND w.duration >= {timeMin} AND w.duration <= {timeMax}

**WITH** w, u, count(r) AS tag**Match**es **ORDER BY** tag**Match**es DESC

**RETURN** w.name, ABS( LENGTH(w.name) - LENGTH({searchWord}) ) AS nameDiff, w.workout\_id, w.dimensions, w.duration, w.difficulty, w.roundCount, w.exCount, w.picPath, u.name, u.user\_id, u.picPath, u.shortBio

**ORDER BY** nameDiff, w.name ASC **SKIP** {**skip**} **LIMIT** {maximum}

**UNION**

**MATCH** (u:User)-[:**CREATE**D]->(w:Workout)-[:HAS\_EXERCISE]->(e:Exercise)-[r:HAS\_TAG]->(t:Tag)

**WHERE** t.name =~ {searchWordRegex} AND w.difficulty >= {difficultyMin} AND w.difficulty <= {difficultyMax} AND w.duration >= {timeMin} AND w.duration <= {timeMax}

**WITH** w, u, count(r) AS tag**Match**es **ORDER BY** tag**Match**es DESC

**RETURN** w.name, ABS( LENGTH(w.name) - LENGTH({searchWord}) ) AS nameDiff, w.workout\_id, w.dimensions, w.duration, w.difficulty, w.roundCount, w.exCount, w.picPath, u.name, u.user\_id, u.picPath, u.shortBio

**ORDER BY** nameDiff, w.name ASC **SKIP** {**skip**} **LIMIT** {maximum}

## neo4j\_getWorkoutExercises

**MATCH** (w:Workout {workout\_id: {workout\_id} })

**WITH** w **MATCH** (w)-[r:HAS\_EXERCISE]->(e:Exercise)

**RETURN** e.exercise\_id, e.name, e.picPath, e.alias, e.equipment, e.execModi, e.dimensions, e.difficulty

## neo4j\_getWorkoutTags

**MATCH** (w:Workout {workout\_id: {workout\_id} })

**WITH** w **MATCH** (w)-[r:HAS\_TAG]->(t:Tag)

**RETURN** t.name

## neo4j\_getRandomExercises

**MATCH** (e)-[:HAS\_TAG]->(t:Tag) **WHERE** t.name IN {tags}

**WITH** e, rand() as rand **ORDER BY** rand **LIMIT** 1000

**RETURN** e.name, e.exercise\_id, e.duration, e.alias, e.picPath, e.shortDescr, e.execModi, e.dimensions, e.equipment, e.difficulty

**SKIP** {pagination} **LIMIT** {maximum}

## neo4j\_getSimilarExercises

**MATCH** (e1:Exercise {exercise\_id: {exercise\_id} })-[:SIMILAR]-(e:Exercise)

**WITH** e

**MATCH** (w:Workout)-[r:HAS\_EXERCISE]->(e)

**RETURN** e.name, e.exercise\_id, e.duration, e.alias, e.picPath, e.shortDescr, e.execModi, e.dimensions, e.equipment, e.difficulty, count(r) as useCases

**ORDER BY** useCases **LIMIT** {maximum}

## neo4j\_searchExercise

**MATCH** (e:Exercise)

**WHERE** e.name =~ {searchWordRegex}

**RETURN** e.name, ABS( LENGTH(e.name) - LENGTH({searchWord}) ) AS nameDiff, e.exercise\_id, e.duration, e.alias, e.picPath, e.shortDescr, e.execModi, e.dimensions, e.equipment, e.difficulty

**ORDER BY** nameDiff, e.name ASC **SKIP** {**skip**} **LIMIT** {maximum}

**UNION**

**MATCH** (e:Exercise)-[r:HAS\_TAG]->(t)

**WHERE** t.name =~ {searchWordRegex}

**WITH** e, count(r) AS tag**Match**es **ORDER BY** tag**Match**es DESC

**RETURN** e.name, ABS( LENGTH(e.name) - LENGTH({searchWord}) ) AS nameDiff, e.exercise\_id, e.duration, e.alias, e.picPath, e.shortDescr, e.execModi, e.dimensions, e.equipment, e.difficulty

**ORDER BY** nameDiff, e.name ASC **SKIP** {**skip**} **LIMIT** {maximum}

## neo4j\_searchUserByName

**MATCH** (u:User)

**WHERE** u.name =~ {searchWordRegex}

**RETURN** u.name, u.user\_id, u.confirmed, u.contact, u.visible, u.picPath, u.shortBio

**ORDER BY** u.name ASC **SKIP** {**skip**} **LIMIT** {maximum}

## neo4j\_getFollowers

**MATCH** (u1:User {user\_id: {user\_id}} )-[:FOLLOWS]->(u:User)

**WITH** u1, u

OPTIONAL **MATCH** (u1)-[:LIKED]->(w:Workout)<-[:LIKED]-(u)

**RETURN** u.name, u.user\_id, u.confirmed, u.contact, u.visible, u.picPath, u.shortBio, count(w) as commonLikes

**ORDER BY** commonLikes **SKIP** {pagination} **LIMIT** {maximum}

## neo4j\_userFollowedUser

**MATCH** (u:User {user\_id: {user\_id} })-[:FOLLOWS]->(u2:User {user\_id: {followed\_id} })

**RETURN** count(\*) <> 0 AS count

## neo4j\_userLikedWorkout

**MATCH** (u:User {user\_id: {user\_id} })-[:LIKED]->(w:Workout {workout\_id: {workout\_id} })

**RETURN** count(\*) <> 0 AS count

## neo4j\_getPWRe**set**Token

**MATCH** (u:User {user\_id: {user\_id} })

**RETURN** u.pwRe**set**Token

# Writing

## neo4j\_**create**Tag

**MERGE** (t:Tag {name: {tagNameInput} })

**RETURN** t.name

## neo4j\_workoutHasTag

**MATCH** (w:Workout)

**WHERE** w.workout\_id = {workout\_id}

**WITH** w **MATCH** (t:Tag)

**WHERE** t.name IN {tagsArray}

**WITH** w, t **MERGE** (w)-[:HAS\_TAG]->(t)

## neo4j\_removeWorkoutHasTag

**MATCH** (w:Workout {workout\_id: {workout\_id} })-[r:HAS\_TAG]->(t:Tag)

**WHERE** t.name IN {tags}

DELETE r

## neo4j\_exerciseHasTag

**MATCH** (e:Exercise)

**WHERE** e.exercise\_id = {exercise\_id}

**WITH** e **MATCH** (t:Tag)

**WHERE** t.name IN {tagsArray}

**WITH** e, t **MERGE** (e)-[:HAS\_TAG]->(t)

## neo4j\_removeExerciseHasTag

**MATCH** (e:Exercise {exercise\_id: {exercise\_id} })-[r:HAS\_TAG]->(t:Tag)

**WHERE** t.name IN {tags}

DELETE r

## neo4j\_**create**Workout

**MATCH** (u:User {user\_id: {user\_id} }) **WITH** u **MERGE** (w:Workout {workout\_id: {workout\_id} })<-[:**CREATE**D { **create**d\_on: TIMESTAMP() } ]-(u)

**ON CREATE** **SET** w = {props}

## neo4j\_workoutHasExercise

**MATCH** (w:Workout {workout\_id: {workout\_id} })

**WITH** w **MATCH** (e:Exercise) **WHERE** e.exercise\_id IN {exercise\_idArray}

**WITH** w, e

**MERGE** (w)-[:HAS\_EXERCISE]->(e)

## neo4j\_removeWorkoutHasExercises

**MATCH** (w:Workout {workout\_id: {workout\_id} })-[r:HAS\_EXERCISE]->(e:Exercise)

**WHERE** e.exercise\_id IN {exercise\_ids}

DELETE r

## neo4j\_updateWorkout

**MATCH** (w:Workout {workout\_id: {workout\_id} })

**SET** w += {props}

**WITH** w **MATCH** (w)<-[:**CREATE**D]-(u:User)

**RETURN** w.name, w.workout\_id, w.dimensions, w.duration, w.roundCount, w.exCount, w.picPath, u.name, u.user\_id, u.picPath, u.shortBio

## neo4j\_removeWorkout

**MATCH** (w:Workout {workout\_id: {workout\_id} })-[r]-()

DELETE r, w

## neo4j\_likeDislikeWorkout

**MATCH** (u:User {user\_id: {user\_id} }), (w:Workout {workout\_id: {workout\_id} })

**WITH** u, w **CREATE** (u)-[:LIKED { **create**d\_on: TIMESTAMP() } ]->(w)

**WITH** u, w **MATCH** (u)-[r:LIKED]->(w), (u)-[:LIKED]->(w)

DELETE r **WITH** u, w **MATCH** (u)-[:LIKED]->(w)

**RETURN** count(\*) = 0 AS count

## neo4j\_**create**User

**MERGE** (u:User {email: {email}})

**ON CREATE** **SET** u = {props}

**RETURN** u.name, u.user\_id

## neo4j\_get3doUser

**MATCH** (u:User) **WHERE** u.confirmed = false AND u.rememberStatus IS NULL AND u.**create**d\_on < {timeEdge}

**RETURN** u.name AS name, u.user\_id AS user\_id, u.email AS email, u.locale AS locale, u.confirmationCode AS confirmationCode

## neo4j\_markRememberedUser

**MATCH** (u:User {u.user\_id = {userID} })

**SET** u.rememberStatus = {rememberStatus}

**RETURN** count(\*) = 1 AS count

## neo4j\_confirmUserMail

**MATCH** (u:User {confirmationCode: {code}})

**SET** u.confirmed = true, u.confirmed\_on = TIMESTAMP()

REMOVE u.confirmationCode

**RETURN** count(\*) = 1 AS count, u.name AS name, u.email AS email, u.locale AS locale

## neo4j\_followUnfollow

**MATCH** (u1:User {user\_id: {user\_id\_1} }), (u2:User {user\_id: {user\_id\_2} })

**WITH** u1, u2 **CREATE** (u1)-[:FOLLOWS { **create**d\_on: TIMESTAMP() } ]->(u2)

**WITH** u1, u2 **MATCH** (u1)-[r:FOLLOWS]->(u2), (u1)-[:FOLLOWS]->(u2)

DELETE r **WITH** u1, u2 **MATCH** (u1)-[:FOLLOWS]->(u2)

**RETURN** count(\*) = 0 AS count

## neo4j\_updateUserProfile

**MATCH** (u:User {user\_id: {user\_id} })

**SET** u += {props}

**RETURN** u.name, u.user\_id, u.email, u.contact, u.visible, u.shortBio, u.locale, u.picPath

## neo4j\_**set**PWRe**set**Token

**MATCH** (u:User {user\_id: {user\_id} })

**SET** u.pwRe**set**Token = {token}

**RETURN** count(\*) <> 0 AS success

## neo4j\_**set**UserPassword

**MATCH** (u:User {user\_id: {user\_id} })

**SET** u.password = {newPassword}

REMOVE u.pwRe**set**Token

**RETURN** count(\*) <> 0 AS success

## neo4j\_removeUser

**MATCH** (u1:User {user\_id: {user\_id} })

**WITH** u1

**MATCH** (w:Workout)<-[r1:**CREATE**D|:LIKED]-(u1)-[r2:FOLLOWS]-(u2:User)

**WITH** w, r1, u1, u2

**MERGE** (w)<-[rNew]-(u3:User {user\_id: 0 }) **SET** rNew = r1

**WITH** w, r1, u1, u2

DELETE r1, r2, u1

## neo4j\_**create**Exercise

**MERGE** (e:Exercise {exercise\_id: {exercise\_id} })

**ON CREATE** **SET** e = {props}

## neo4j\_exerciseSimilar

**MATCH** (e1:Exercise), (e2:Exercise)

**WHERE** e1.exercise\_id = {exercise\_id} AND e2.exercise\_id IN {exerciseIdArray}

**WITH** e1, e2

**MERGE** (e1)-[:HAS\_TAG]->(e2)

## neo4j\_removeExerciseSimilar

**MATCH** (e1:Exercise {exercise\_id: {exercise\_id} })-[r:HAS\_TAG]->(e2:Exercise)

**WHERE** e2.exercise\_id IN {exerciseIdArray}

DELETE r

## neo4j\_updateExercise

**MATCH** (e:Exercise {exercise\_id: {exercise\_id} })

**SET** e += {props}

**RETURN** e.name, e.exercise\_id, e.duration, e.alias, e.picPath, e.shortDescr, e.execModi, e.dimensions, e.equipment, e.difficulty

# Metrics

## neo4j\_basicMetrics

**MATCH** (u0:User {confirmed: true}) **WITH** count(u0) AS activeUsers

**MATCH** (u1:User) **WITH** activeUsers, count(u1) AS totalUsers

**MATCH** (w0:Workout) **WITH** activeUsers, totalUsers, count(w0) AS totalWorkouts

**MATCH** (u2:User)-[r0:LIKED]->() **WITH** activeUsers, totalUsers, totalWorkouts, count(r0) AS totalLikes

**MATCH** (u3:User)-[r1:**CREATE**D]->() **WITH** activeUsers, totalUsers, totalWorkouts, totalLikes, u3.name AS mostActiveCname, u3.email AS mostActiveCemail, count(r1) AS mostActiveCworkouts **ORDER BY** mostActiveCworkouts DESC **LIMIT** 1

**MATCH** (u4:User)-[:**CREATE**D]->(w1:Workout)<-[r2:LIKED]-() **RETURN** activeUsers, totalUsers, totalWorkouts, totalLikes, mostActiveCname, mostActiveCemail, mostActiveCworkouts, w1.name AS mostLikedWname, u4.name AS mostLikedWcname, u4.email AS mostLikedWcemail, count(r2) AS mostLikedWLikes **ORDER BY** mostLikedWLikes DESC **LIMIT** 1