COS221 - Practical 5

NOT_NULL_CREW

Jonelle Coertze
u21446271

Wian Koekemoer
u19043512

Arno Jooste
u21457451

Reuben Jooste
u21457060

Jake Weatherhead
u04929552

Themba Mdluli
u19234806

Contents

1 Task 1: Research and Analysis of Golf

1.1 General Overview And Explanation Of Golf

Golf is a club-and-ball sport where players use various clubs to hit balls into a series of holes on a course in as few strokes as possible. Strokes are how golfers advance the ball around the golf course, and each stroke is counted as part of keeping score. However, some strokes are not counted:

- If a golfer completes a swing but intentionally misses the golf ball, that does not count as a stroke. Such a situation might arise from a last minute distraction, perhaps from a spectator or from the weather conditions.
- If a golfer stops their swing before contacting the ball, this does not count as a stroke.
- Depending on the intent of the player, it is possible to miss the golf ball and still have to count that miss as a stroke. If the player changed the course of their swing intentionally, no stroke is counted. However if the player was trying to hit the ball and misses unintentionally, a stroke is recorded.
- There is a penalty stroke incurred by the player for the violation of certain rules as governed by the golfing association in charge of the tournament being played.
- There is, also, the concept of a handicap stroke whereby a player, perhaps of lesse rability, is entitled to additional strokes on a hole without affecting their overall score, as a way of levelling the playing field.

Par is the term used to describe the number of strokes that a golfer with a zero-handicap will require to finish a specific hole. Every hole on every golf course has a par value. Players can take their final score and compare it to the par for the course being played. For example, if a player scores 68 on a par-72 course, they will have a score of 4-under-par, often stylised as -4. Likewise, if they score 76, that would be 4-over-par, or +4.

In serious stroke play competitions, where large prize-pools are involved, golfers are required to finish every hole, regardless of the number of strokes this may take. In handicap competitions, players subtract their handicap from their total gross score and judge their resultant score against the par for the course. For example, if a golfer has a handicap of 12, they deduct 12 strokes from their final score at the end of the round. A gross score of 84 would therefore produce a final score of 72.

There are three different formats of golf that can be played:

- **Stroke play:** the winner is the golfer who uses the fewest number of strokes for the full round of golf (normally 18 holes).
- **Match play:** the winner of the match is the golfer who achieves the lowest score on the most holes.
- **Stabelford:** the winner is the golfer with the most points at the end of the round, as the number of strokes a golfer uses on each hole is converted into points earned.

1.2 Golf is a single-player sport

The game can be played by any number of people, although, typically a group of 2-4 players will play each hole together. The normal amount of time required for pace of play for a nine hole round is two hours, and for an 18-hole round it is four hours. Even though any number of people can play, everyone is a separate entity, which means it is a single player sport played by several people in tandem. During international tournaments, the elite golfers play as single entities and normally tee off in groups of two.

1.3 Actions in the sport of Golf

For every player, every stroke can be seen as an action. The result of the stroke can be captured in terms of distance, club used and where the ball falls (on the fairway, in the rough or on the putting-green).

Additionally, the score per player, per hole, per tournament will be kept as well as the leader board at the end of each round per tournament.

At the tour-level, various statistics can be derived, for example:

1. Most Rounds Below 65
2. Top-10 Finishes
3. Scoring Averages
4. Average Driving Distance Off The Tee
5. Putting-Greens Reached in Regulation
6. Fairways Hit Off The Tee

1.4 Location / Date / Time

Golf events normally take place on a single, physical golf course over four days. Each day, every golfer must play one round (18 holes) of golf. We can take the PGA Championship for the 2021-2022 season as an example which takes place in Tulsa, Oklahoma on May 19-22, 2022 in Southern Hills Course One. This championship has four rounds taking place from 1pm to 7pm Eastern time over four days resulting in a total of 72 holes.

1.5 Sport Structure

There are numerous golfing federations around the world, but the governing body of world golf is the International Golf Federation (IGF) and is the recognised International Federation within the Olympic and Paralympic Movement. There are 150 members of the IGF in 147 countries. More than 60 million players are registered at these federations. Among these is GOLFRSA, the main governing body of golf in South Africa.

During a golfing season, various tournaments are hosted by these members. These tournaments can be single events or multiple events as part of a tour, such as the Professional Golfers' Association (PGA) tour in the United States, and the Sunshine Tour in South Africa.

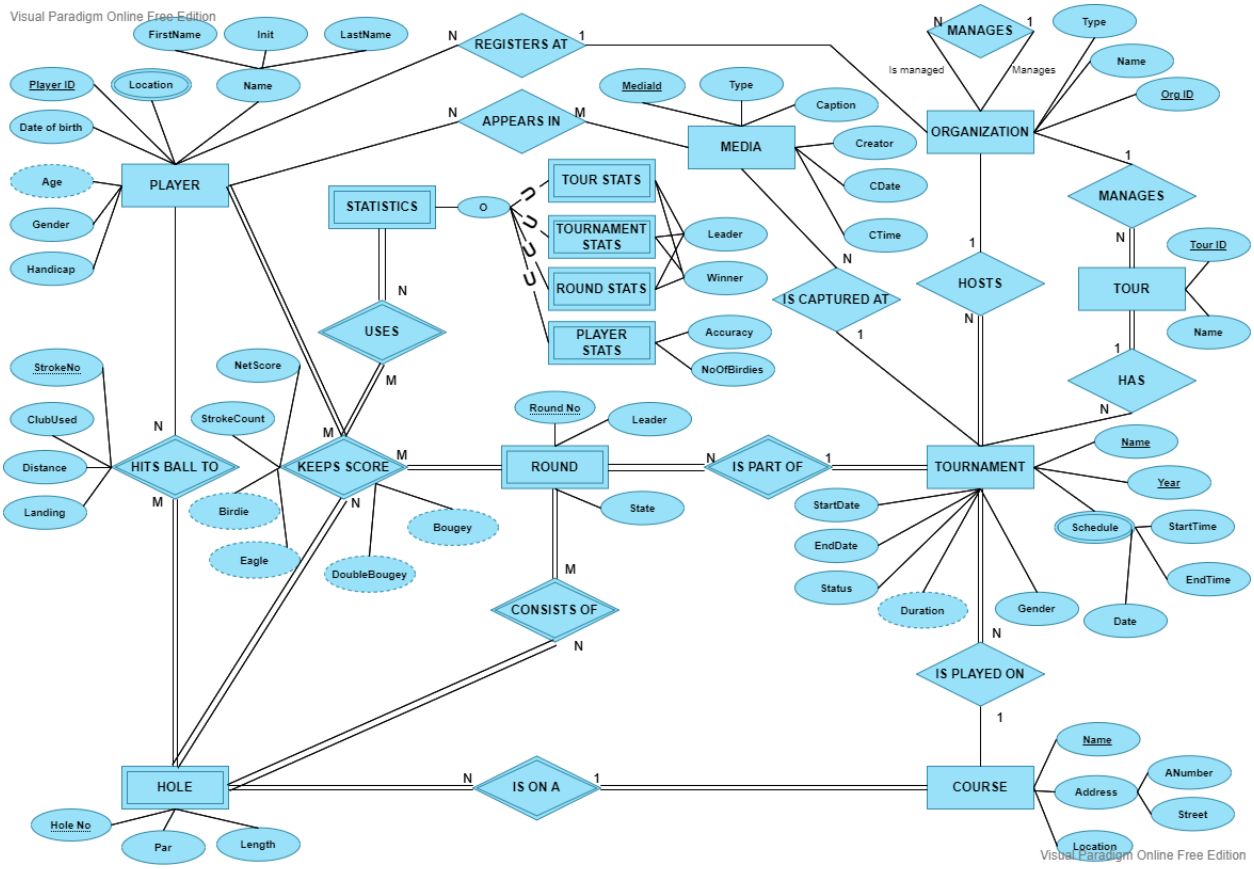
As previously mentioned, in most tournaments, players are seen as single entities, and not as teams, and so the concept of a team generally does not exist in golf. Each tournament takes place at a single golf course, although the various tournaments of a golf tour will take place at different locations.

At the end of a tournament, the winner is the player who completes all the holes in the round in the fewest number of shots, thereby achieving the lowest score.

2 Task 2: (E)ER-Diagram - Iterations And Comments

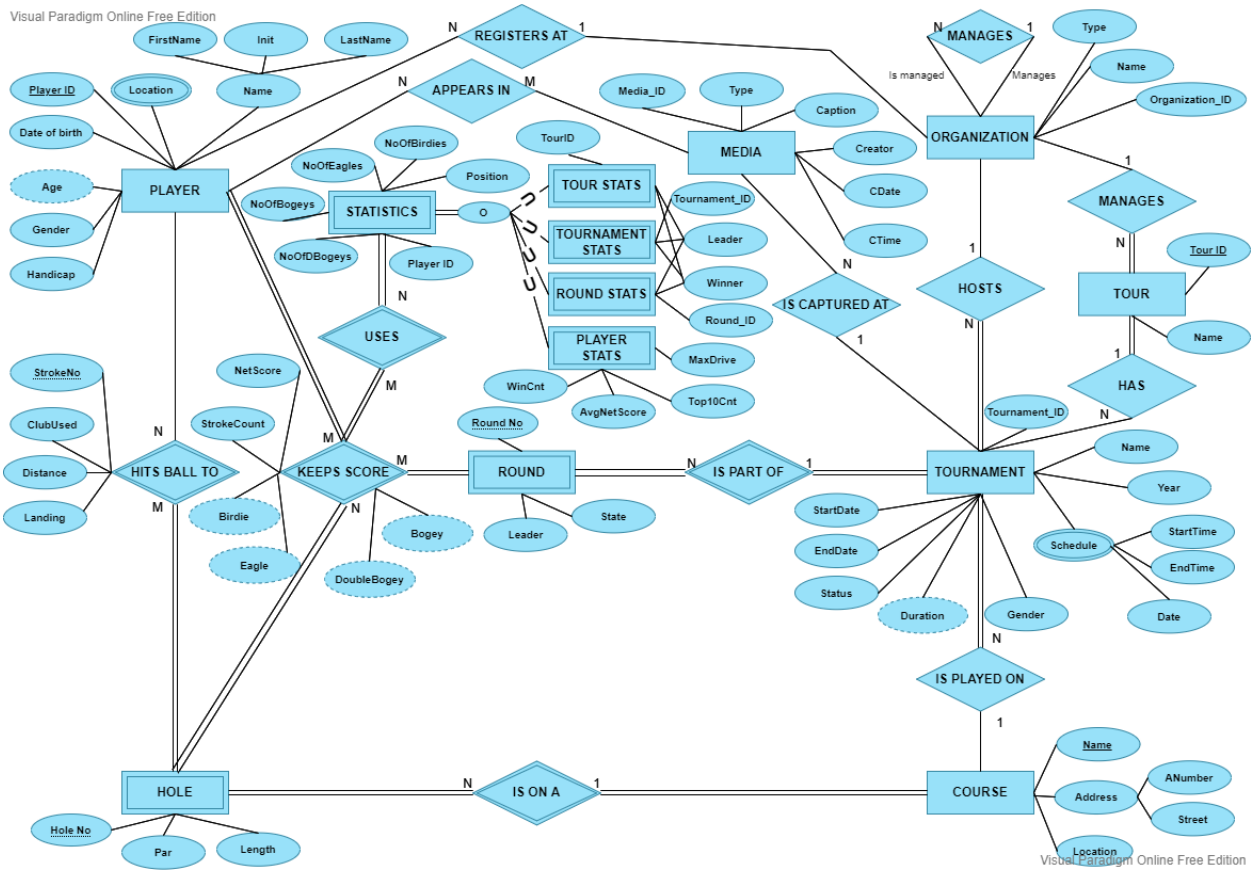
2.0.1 1st Iteration

After an initial assessment of the specifications, we produced the first iteration of our (E)ER-diagram (depicted below).



2.0.2 2nd Iteration

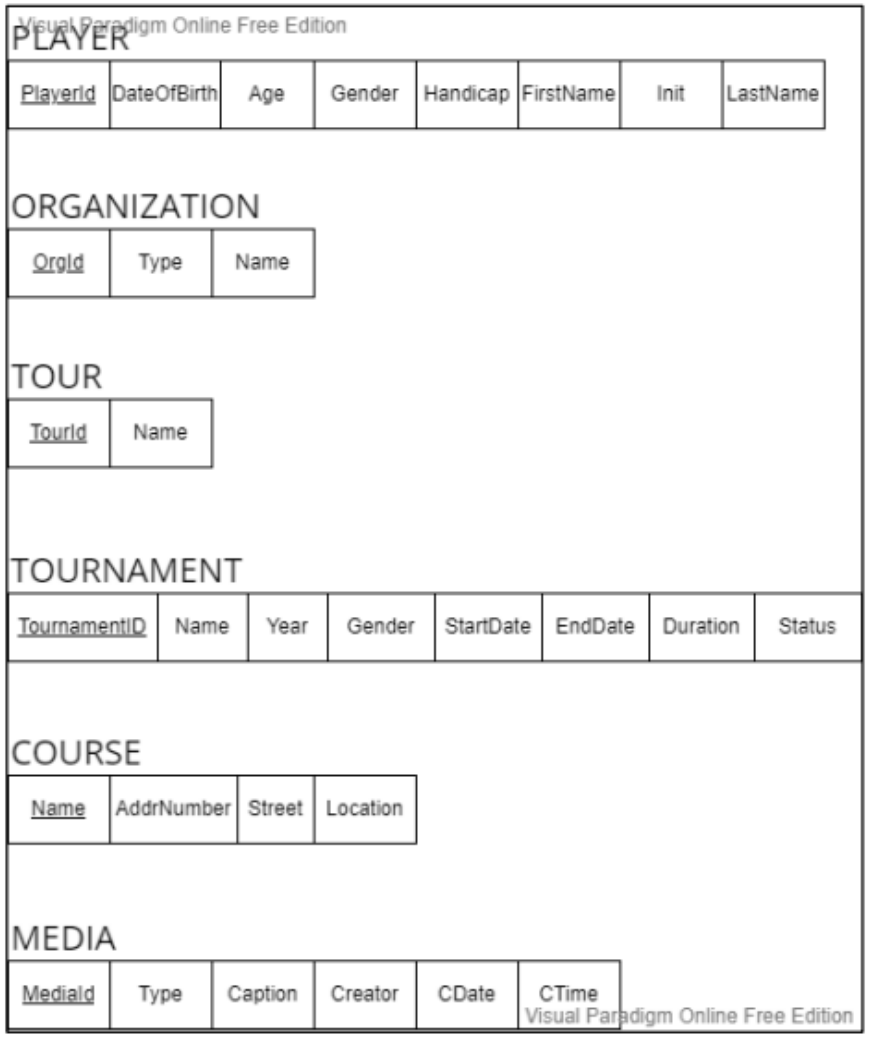
During the conversion of our (E)ER-diagram to a relational mapping, we realized that it would be best to add the tournamentId attribute to the TOURNAMENT entity in order to reduce the number of fields in other relations. We also realized that the relationship, CONSISTS_OF between ROUND and HOLE is not necessary because the KEEPS_SCORE relationship already contains this relationship. Lastly, we realized that our statistics relations did not contain enough attributes, and so we expanded upon the existing relations to capture more detail.



3 Task 3: Relational Mapping

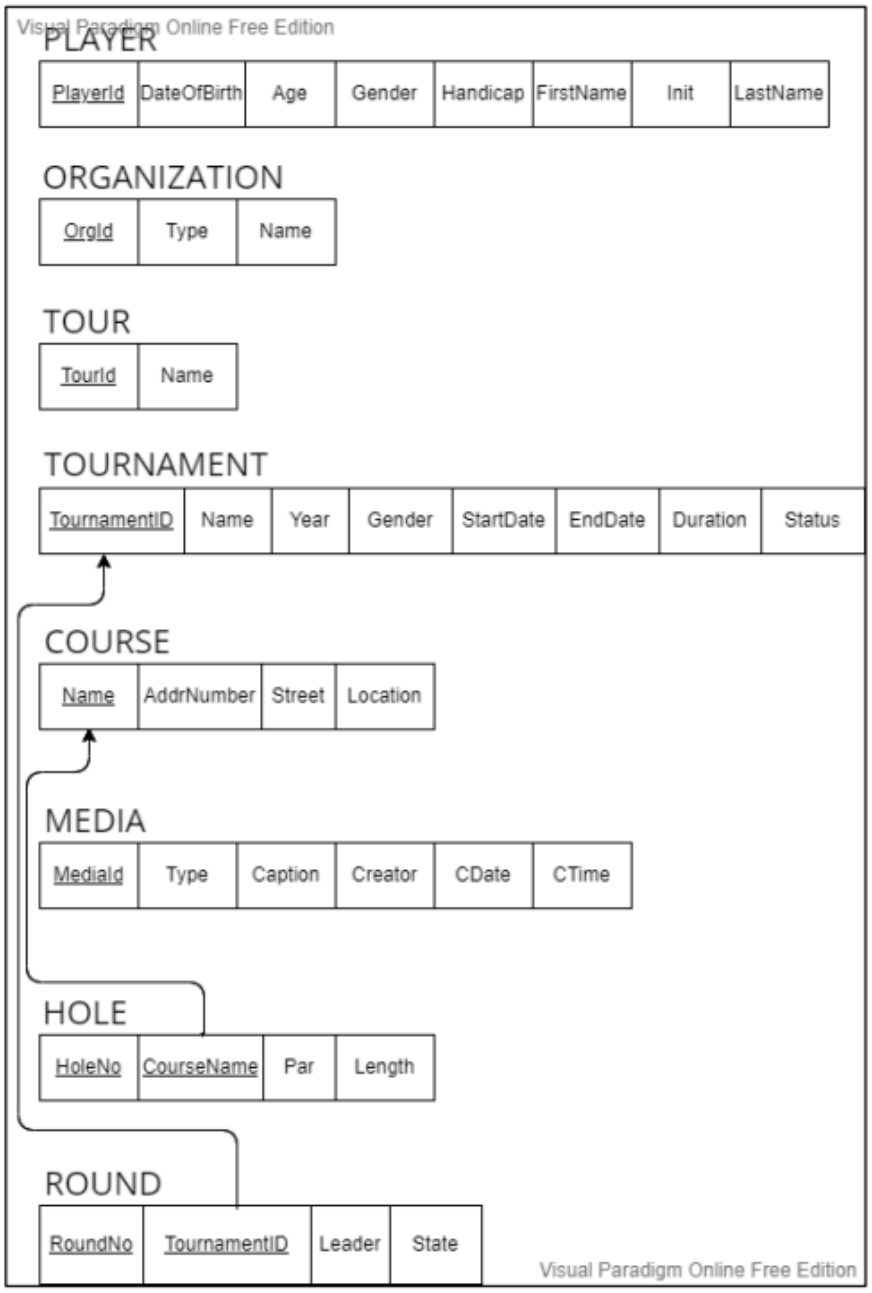
3.1 Step 1: Mapping Of Regular Entity Types

In step 1, we mapped all the strong entity types and included their simple attributes, including the simple attributes of composite attributes.



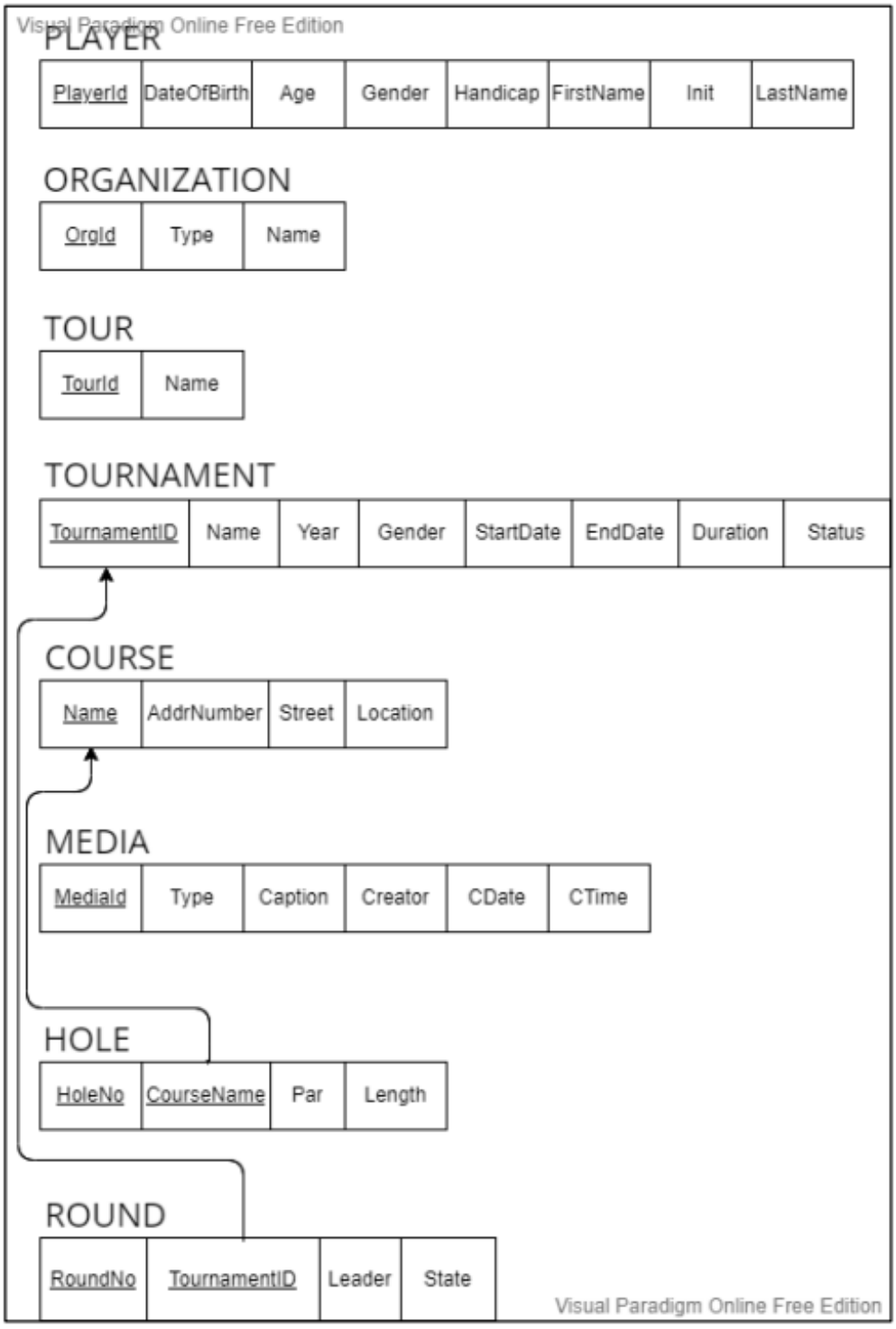
3.2 Step 2: Mapping Of Weak Entity Types

In step 2, we mapped the weak entities along with their simple attributes.



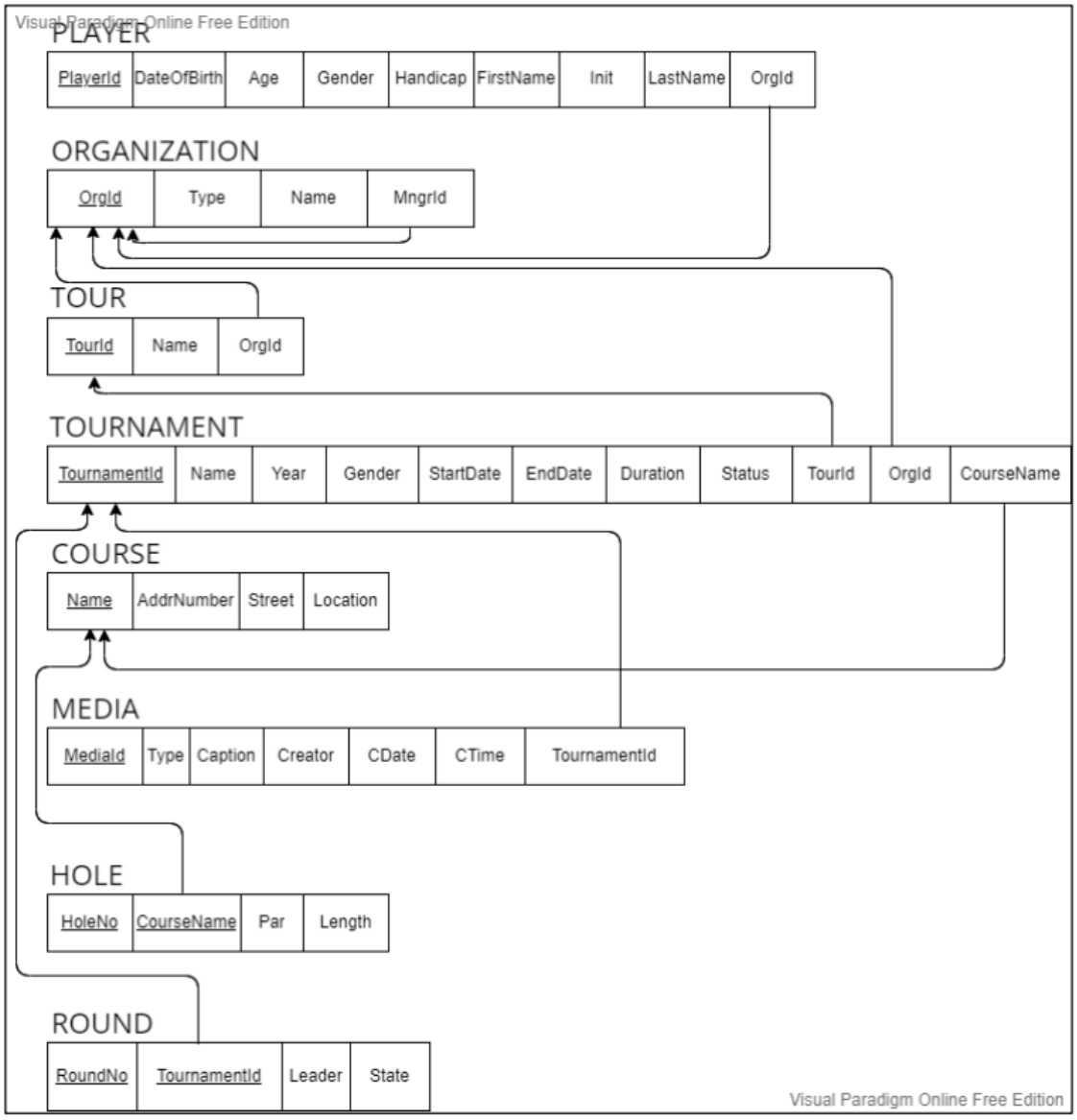
3.3 Step 3: Mapping Of Binary 1:1 Relationship Types

In step 3, our diagram stayed the same because we do not have any 1:1 relationships.



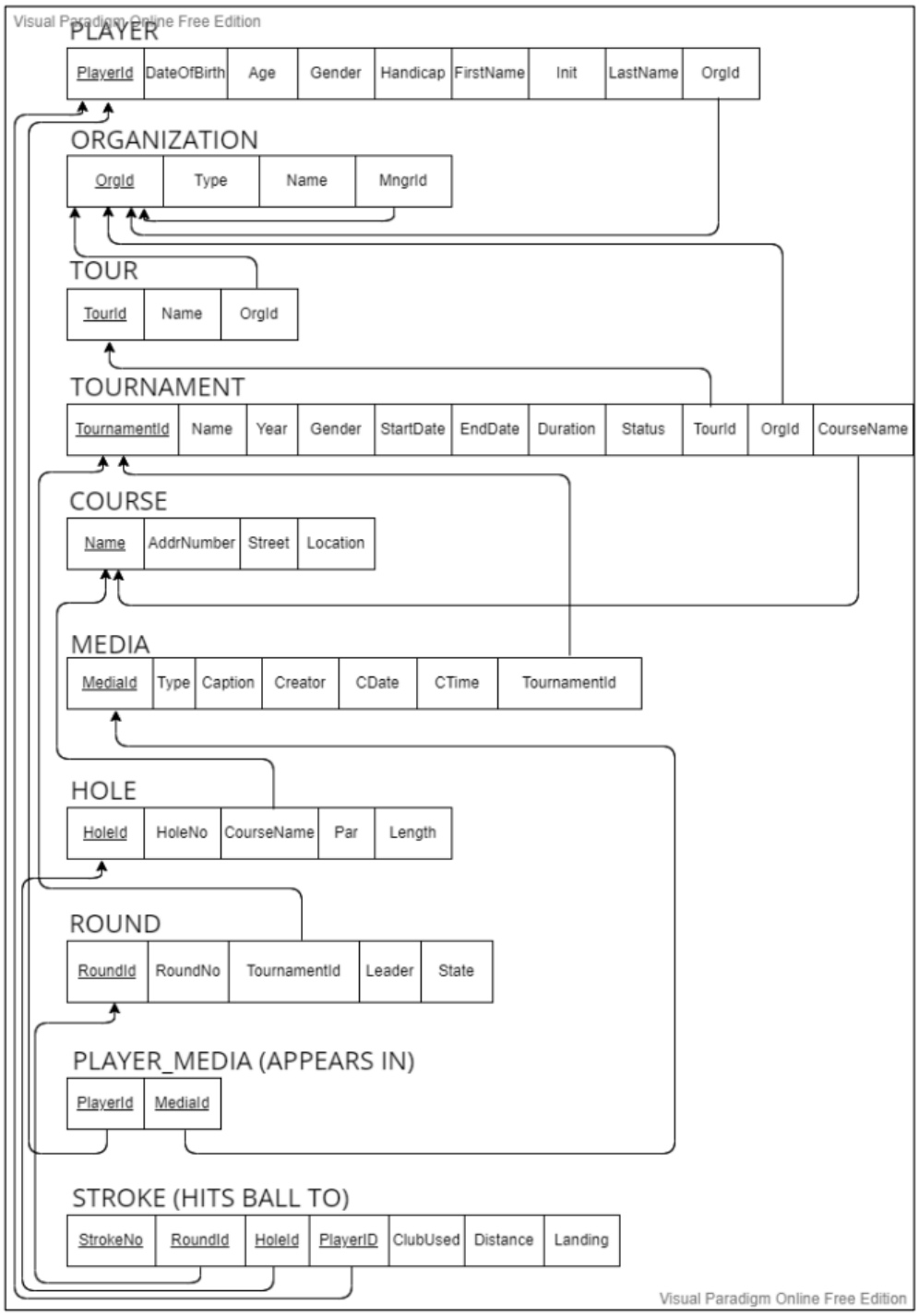
3.4 Step 4: Mapping Of Binary 1:N Relationship Types

In step 4, we added all the 1:N Relationships. For all these relationships, we decided to use the foreign key approach, so we added the primary key of the relation on the 1 side to the relation on the N side as a foreign key.



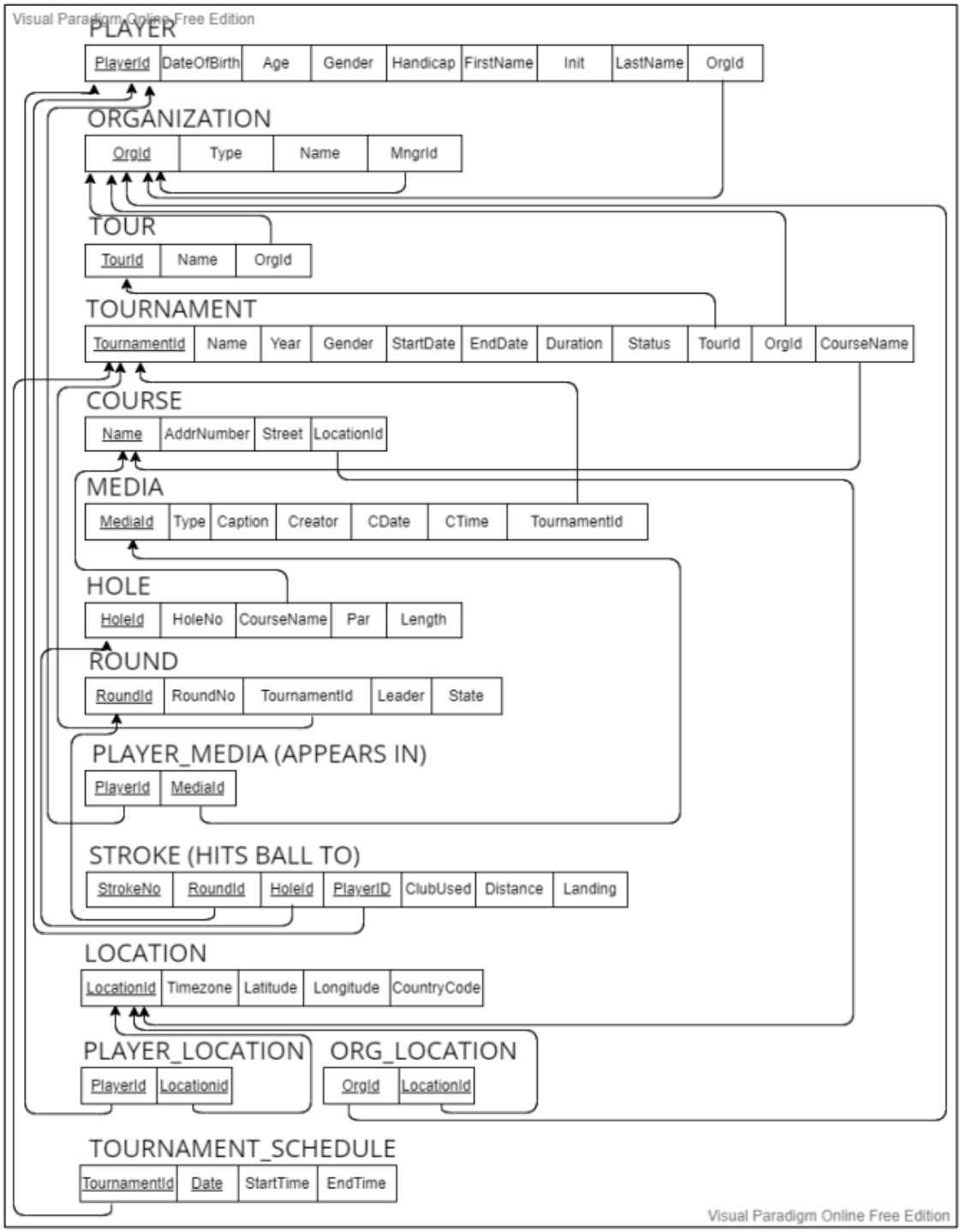
3.5 Step 5: Mapping Of Binary M:N Relationship Types

In step 5, we added all the M:N relationships as new relations. We added a unique ID to each weak entity type of step 2 to ensure that we do not have redundant data.



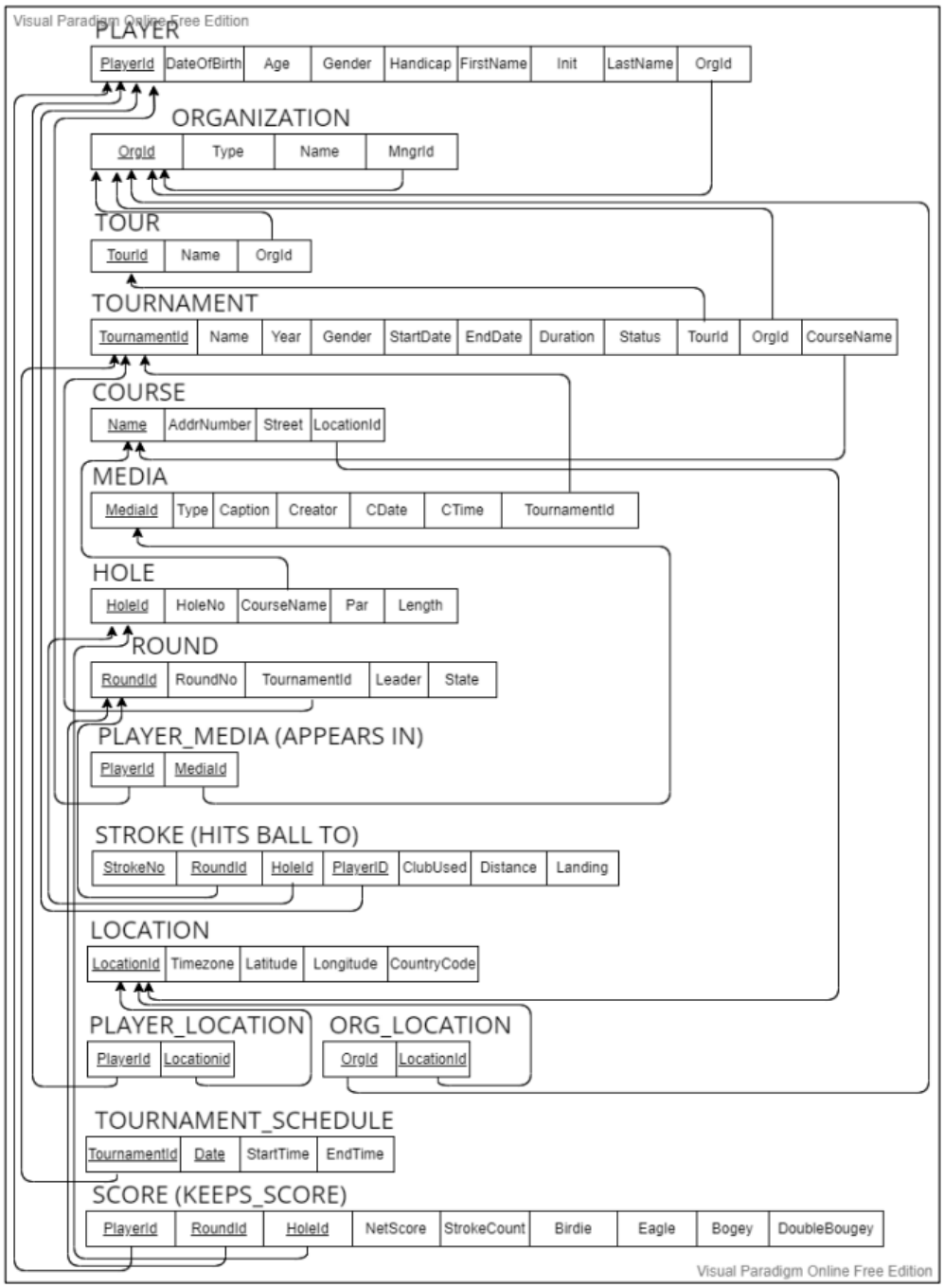
3.6 Step 6: Mapping Of Multivalued Attributes

In step 6, we have 'location' in a few different entities and we decided to go with the same approach as the SportsDB. Thus, LOCATION is a relation on its own and then we also incorporated it into our multivalued attribute relations



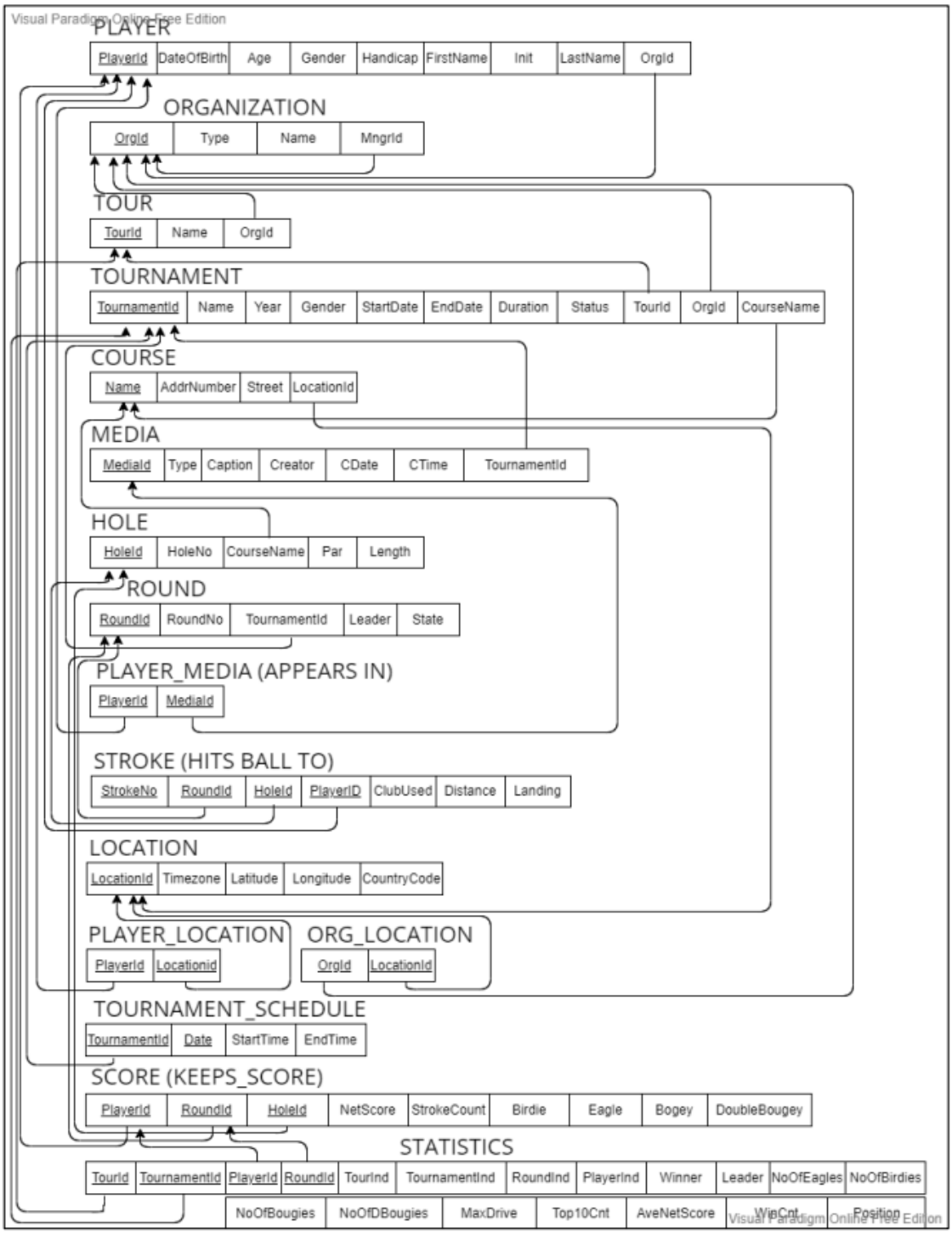
3.7 Step 7: Mapping Of N-ary Relationship Types

In step 7, we have only 1 3-ary relationship to keep score. We decided to create a new relation for that relationship.



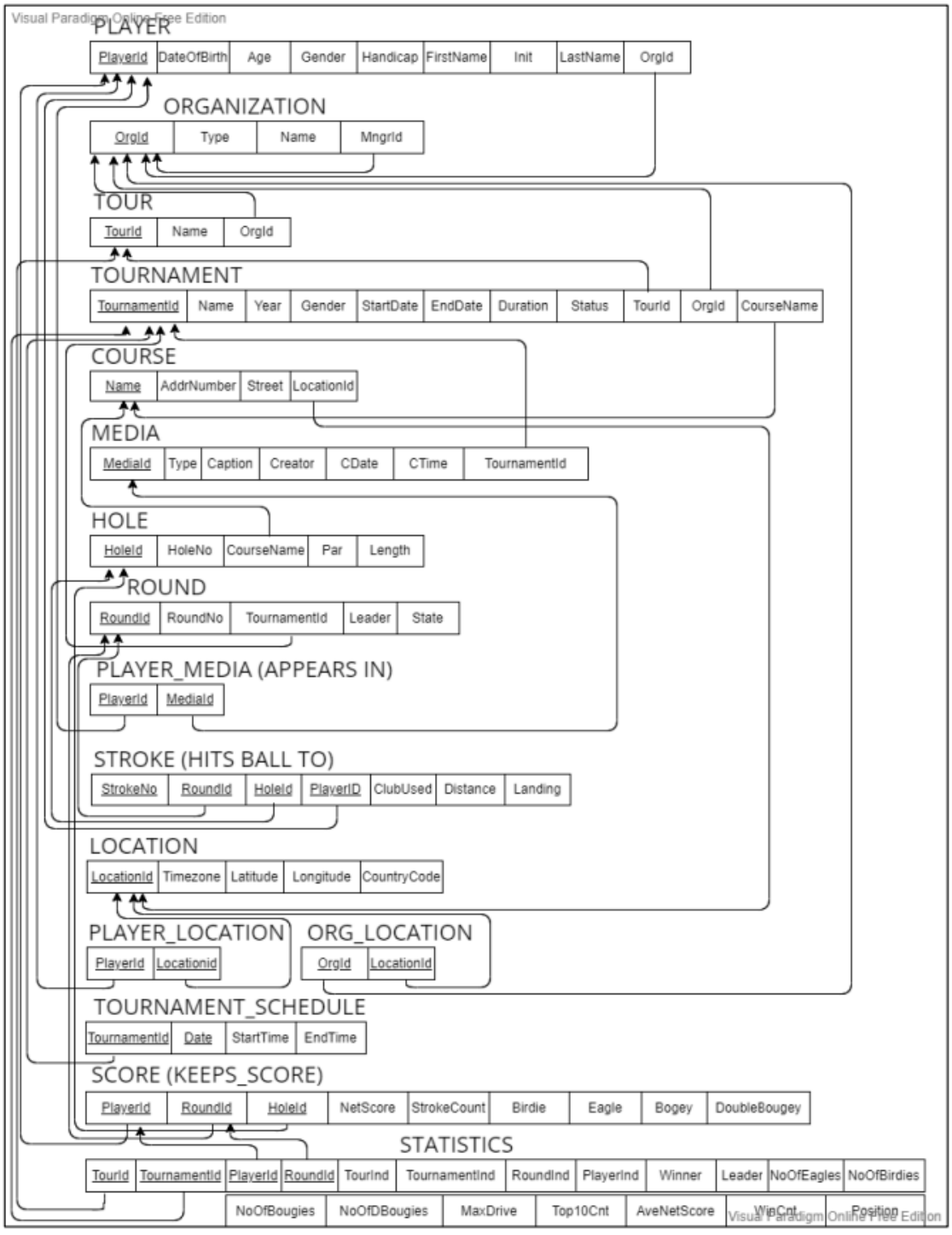
3.8 Step 8: Mapping Specialization or Generalization

In step 8, we have one specialization named STATISTICS. We added the statistics with its subclasses into a single relation with multiple type attributes because we have an overlapping specialization.



3.9 Step 9: Mapping Of Union Types (Categories)

In step 9, our diagram stayed the same because we do not have any union types.



4 Task 4: Relational Exclusion

We used the following assumptions when creating our database:

- We are allowed to add columns to existing SportsDB tables – as confirmed by our mentor Maryam.
- For the visual diagram, we used the format of task 3 and added all other necessary information – as confirmed by our mentor Maryam.
- We do not have to use all the tables in the SportsDB.

4.1 Map Relational Model To SportsDB

We mapped our relation model to the SportsDB and added columns and new tables as needed:

4.1.1 Users Relation

Relation	Attribute		SportDB Table	SportDB Field	SportsDB Table	SportDB New field	New Table	New Field
Users								
	UserId		publishers	`id`			users	id
	Password		None				users	password
	Email		publishers	`publisher_key`			users	email
	Tel_no		None				users	tel_no
	Type		None				users	user_type
	FirstName		None				users	first_name
	Init		None				users	init
	Lastname		None				users	last_name

4.1.2 Player, Organization and Tour Relations

Relation	Attribute		SportDB Table	SportDB Field	SportsDB Table	SportDB New field	New Table	New Field
Player	Player_ID		persons	`id`				
	Date_of_birth		persons	`birth_date`				
	Age		None		persons	age		
	Gender		persons	`gender`				
	Handicap		None		persons	handicap		
	FirstName		display_names	`first_name`				
	Init		display_names	`middle_name`				
	LastName		display_names	`last_name`				
	Organization_ID		None		persons	affiliation_id		
Organization	Organization_ID		affiliations	`id`				
	Type		affiliations	`affiliation_type`				
	Name		affiliations	`affiliation_key`				
	Manager_ID		None		affiliations	manager_id		
Tour	Tour_ID		None				tours	id
	Name		None				tours	tour_name
	Organization_ID		None				tours	affiliation_id

4.1.3 Tournament, Course and Media Relations

Relation	Attribute		SportDB Table	SportDB Field	SportsDB Table	SportDB New field	New Table	New Field
Tournament	Tournament_ID		events	`id`				
	Name		events	`event_key`				
	Year		None		events	year		
	Gender		None		events	gender		
	StartDate		events	`start_date_time`				
	EndDate		events	`end_date_time`				
	Duration		events	`duration`				
	Status		events	`event_status`				
	Tour_ID		None		events	tour_id		
	Organization_ID		affiliation_events	affiliation_id				
	CourseName		events	`site_id`				
Course	Name		sites	`site_key`				
	ANumber		addresses	`street_number`	sites	address_id		
	Street		addresses	`street`				
	Location_ID		sites	`location_id`				
Media	Media_ID		media	`id`				
	Type		media	`media_type`				
	Caption		media_captions	`caption`				
	Creator		media	`credit_id`				
	Cdate		media	`date_time`				
	CTIme		media	`date_time`				
	Tournament_ID		events_media	`event_id`				

4.1.4 Hole, Round, Player_Media and Stroke Relations

Relation	Attribute		SportDB Table	SportDB Field	SportsDB Table	SportDB New field	New Table	New Field
Hole	Hole_ID		None				holes	id
	Hole_No		None				holes	hole_no
	CourseName		None				holes	site_id
	Par		None				holes	par
	Length		None				holes	length
Round	Round_ID		None				rounds	id
	Round_No		None				rounds	round_no
	Tournament_ID		None				rounds	event_id
	Leader		None				rounds	leader_id
	State		None				rounds	state
Player_Media	Player_ID		persons_media	`person_id`				
	Media_ID		persons_media	`media_id`				
Stroke	StrokeNo		None				strokes	stroke_no
	Round_ID		None				strokes	round_id
	Hole_ID		None				strokes	hole_id
	Player_ID		None				strokes	person_id
	ClubUsed		None				strokes	club_used
	Distance		None				strokes	distance
	Landing		None				strokes	landing

4.1.5 Location, Player_Location and Tournament_Schedule Relations

Relation	Attribute		SportDB Table	SportDB Field	SportsDB Table	SportDB New field	New Table	New Field
Location	Location_ID		locations	`id`				
	City		locations	`city`				
	Timezone		locations	`timezone`				
	Latitude		locations	`latitude`				
	Longitude		locations	`longitude`				
	CountryCode		locations	`country`				
Player_Location								
	Player_ID		persons	`id`				
	Location_ID		persons	`birth_location_id`				
Tournament_Schedule								
	ID		None				tournament_schedule	id
	Tournament_ID		None				tournament_schedule	event_id
	Date		None				tournament_schedule	date
	StartTime		None				tournament_schedule	start_time
	EndTIme		None				tournament_schedule	end_time

4.1.6 Score Relation

Relation	Attribute		SportDB Table	SportDB Field	SportsDB Table	SportDB New field	New Table	New Field
Score								
	Player_ID		None				scores	person_id
	Round_ID		None				scores	round_id
	Hole_ID		None				scores	hole_id
	NetScore		None				scores	net_score
	StrokeCount		None				scores	stroke_count
	Birdie		None				scores	birdie
	Eagle		None				scores	eagle
	Bogey		None				scores	bogey
	DoubleBogey		None				scores	double_bogey

4.1.7 Statistics Relation

Relation	Attribute		SportDB Table	SportDB Field	SportsDB Table	SportDB New field	New Table	New Field
Statistics								
	ID		None				golf_statistics	id
	Type		None				golf_statistics	entity_type
	Tour_ID		None				golf_statistics	entity_id
	Tournament_ID		None				golf_statistics	entity_id
	Player_ID		None				golf_statistics	entity_id
	Round_ID		None				golf_statistics	entity_id
	TourInd		None				golf_statistics	tour_ind
	TournamentInd		None				golf_statistics	event_ind
	RoundInd		None				golf_statistics	round_ind
	PlayerInd		None				golf_statistics	player_ind
	Winner		None				golf_statistics	winner_id
	Leader		None				golf_statistics	leader_id
	NoOfEagles		None				golf_statistics	no_of_eagles
	NoOfBirdies		None				golf_statistics	no_of_birdies
	NoOfBogeys		None				golf_statistics	no_of_bogeys
	NoOfDBogeys		None				golf_statistics	no_of_double_bogeys
	MaxDrive		None				golf_statistics	max_drive
	Top10Cnt		None				golf_statistics	top10_cnt
	AvgNetScore		None				golf_statistics	avg_net_score
	WinCnt		None				golf_statistics	win_cnt
	Position		None				golf_statistics	position

4.2 SQL to add changes to the database

- Create new table USERS:

```
CREATE TABLE ‘users‘ (
    ‘id‘ int(11) NOT NULL AUTO_INCREMENT,
    ‘email‘ varchar(320) NOT NULL,
    ‘password‘ varchar(100) NOT NULL,
    ‘tel_no‘ char(10) NOT NULL,
    ‘user_type‘ varchar(6) NOT NULL,
    ‘first_name‘ varchar(100) NOT NULL,
    ‘init‘ varchar(5) DEFAULT NULL,
    ‘last_name‘ varchar(100) NOT NULL,
    PRIMARY KEY (‘id‘),
    UNIQUE KEY ‘email‘ (‘email‘),
    CONSTRAINT ‘CHK_tel‘ CHECK (‘tel_no‘ regexp ’0[0-9]{9}’),
    CONSTRAINT ‘CHK_type‘ CHECK (‘user_type‘ regexp ’(admin|normal)’)
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8mb4;
```

- Add one new column ADDRESS_ID to table SITES:

  - ALTER TABLE sites ADD COLUMN address_id INT(11) NOT NULL;
  - ALTER TABLE sites ADD CONSTRAINT fk_address_id
  - FOREIGN KEY (address_id) REFERENCES ADDRESSES (id);

- Create new table HOLES:

```
CREATE TABLE ‘holes‘ (
    ‘id‘ int(11) NOT NULL AUTO_INCREMENT,
    ‘hole_no‘ int(2) NOT NULL,
    ‘site_id‘ int(11) NOT NULL,
    ‘par‘ int(2) NOT NULL,
    ‘length‘ decimal(10,2) NOT NULL,
    PRIMARY KEY (‘id‘),
    UNIQUE KEY ‘hole_no‘ (‘hole_no‘,‘site_id‘),
    KEY ‘site_id‘ (‘site_id‘),
    CONSTRAINT ‘holes_ibfk_1‘ FOREIGN KEY (‘site_id‘) REFERENCES ‘sites‘ (‘id‘),
    CONSTRAINT ‘CHK_hole_no‘ CHECK (‘hole_no‘ > 0),
    CONSTRAINT ‘CHK_par‘ CHECK (‘par‘ > 0),
    CONSTRAINT ‘CHK_length‘ CHECK (‘length‘ > 0)
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8mb4;
```

- Add attiribute MANAGER_ID to table AFFILIATIONS:

  - ALTER TABLE affiliations ADD COLUMN manager_id INT(11);
  - ALTER TABLE affiliations ADD CONSTRAINT fk_manager_id
    FOREIGN KEY (manager_id) REFERENCES AFFILIATIONS (id);

- Create new table TOURS:

```
CREATE TABLE ‘tours‘ (
    ‘id‘ int(11) NOT NULL AUTO_INCREMENT,
    ‘tour_name‘ varchar(100) NOT NULL,
    ‘affiliation_id‘ int(11) NOT NULL,
    PRIMARY KEY (‘id‘),
    UNIQUE KEY ‘tour_name‘ (‘tour_name‘),
    KEY ‘FK_tours_aff_id__aff_id‘ (‘affiliation_id‘),
    CONSTRAINT ‘FK_tours_aff_id__aff_id‘ FOREIGN KEY (‘affiliation_id‘) REFERENCES
     ‘affiliations‘ (‘id‘)
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8mb4;
```

- Add three new columns to table EVENTS:

  - ALTER TABLE events ADD COLUMN gender set(‘Men‘, ‘Women‘) NOT NULL;
  - ALTER TABLE events ADD COLUMN tour_id INT(11);
  - ALTER TABLE events ADD COLUMN year INT(4) NOT NULL;
  - ALTER TABLE events ADD CONSTRAINT fk_tour_id
    FOREIGN KEY (tour_id) REFERENCES TOURS (id);

- Create new table ROUNDS:

```
CREATE TABLE ‘rounds‘ (
    ‘id‘ int(11) NOT NULL AUTO_INCREMENT,
    ‘round_no‘ int(2) NOT NULL,
    ‘event_id‘ int(11) NOT NULL,
    ‘leader_id‘ int(11) DEFAULT NULL,
    ‘state‘ set(’Scheduled’,’In progress’,’Finished’,’Cancelled’,’Postponed’) NOT NULL,
    PRIMARY KEY (‘id‘),
    UNIQUE KEY ‘round_no‘ (‘round_no‘,‘event_id‘),
    KEY ‘FK_rounds_event_id__events_id‘ (‘event_id‘),
    KEY ‘FK_rounds_leader_id__persons_id‘ (‘leader_id‘),
    CONSTRAINT ‘FK_rounds_event_id__events_id‘ FOREIGN KEY (‘event_id‘) REFERENCES
     ‘events‘ (‘id‘),
    CONSTRAINT ‘FK_rounds_leader_id__persons_id‘ FOREIGN KEY (‘leader_id‘) REFERENCES
     ‘persons‘ (‘id‘)
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8mb4;
```

- Create new table TOURNAMENT_SCHEDULE:

```
CREATE TABLE ‘tournament_schedules‘ (
    ‘id‘ int(11) NOT NULL AUTO_INCREMENT,
    ‘event_id‘ int(11) NOT NULL,
    ‘date‘ date NOT NULL,
    ‘start_time‘ time NOT NULL,
    ‘end_time‘ time NOT NULL,
    PRIMARY KEY (‘id‘),
    UNIQUE KEY ‘event_id‘ (‘event_id‘,‘date‘),
    CONSTRAINT ‘FK_tourn_sched_event_id__events_id‘ FOREIGN KEY (‘event_id‘)
     REFERENCES ‘events‘ (‘id‘),
    CONSTRAINT ‘CHK_time‘ CHECK (‘start_time‘ < ‘end_time‘)
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8mb4;
```

- Add three new columns to table PERSONS:

  - ALTER TABLE persons ADD COLUMN age INT(4) NOT NULL;
  - ALTER TABLE persons ADD COLUMN handicap INT(4);
  - ALTER TABLE persons ADD COLUMN affiliation_id INT(11) NOT NULL;
  - ALTER TABLE persons ADD CONSTRAINT fk_affiliation_id
    FOREIGN KEY (affiliation_id) REFERENCES AFFILIATIONS (id);

- Create new table SCORES:

```
CREATE TABLE ‘scores‘ (
    ‘person_id‘ int(11) NOT NULL,
    ‘round_id‘ int(11) NOT NULL,
    ‘hole_id‘ int(11) NOT NULL,
    ‘net_score‘ int(3) DEFAULT NULL,
    ‘stroke_count‘ int(2) DEFAULT NULL,
    ‘birdie‘ tinyint(1) DEFAULT NULL,
    ‘eagle‘ tinyint(1) DEFAULT NULL,
    ‘bogey‘ tinyint(1) DEFAULT NULL,
    ‘double_bogey‘ tinyint(1) DEFAULT NULL,
    PRIMARY KEY (‘person_id‘,‘round_id‘,‘hole_id‘),
    KEY ‘FK_scores_round_id__rounds_id‘ (‘round_id‘),
    KEY ‘FK_scores_hole_id__holes_id‘ (‘hole_id‘),
    CONSTRAINT ‘FK_scores_hole_id__holes_id‘ FOREIGN KEY (‘hole_id‘) REFERENCES
     ‘holes‘ (‘id‘),
    CONSTRAINT ‘FK_scores_person_id__persons_id‘ FOREIGN KEY (‘person_id‘) REFERENCES
     ‘persons‘ (‘id‘),
    CONSTRAINT ‘FK_scores_round_id__rounds_id‘ FOREIGN KEY (‘round_id‘) REFERENCES
     ‘rounds‘ (‘id‘),
    CONSTRAINT ‘CHK_strokes‘ CHECK (‘stroke_count‘ >= 0)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

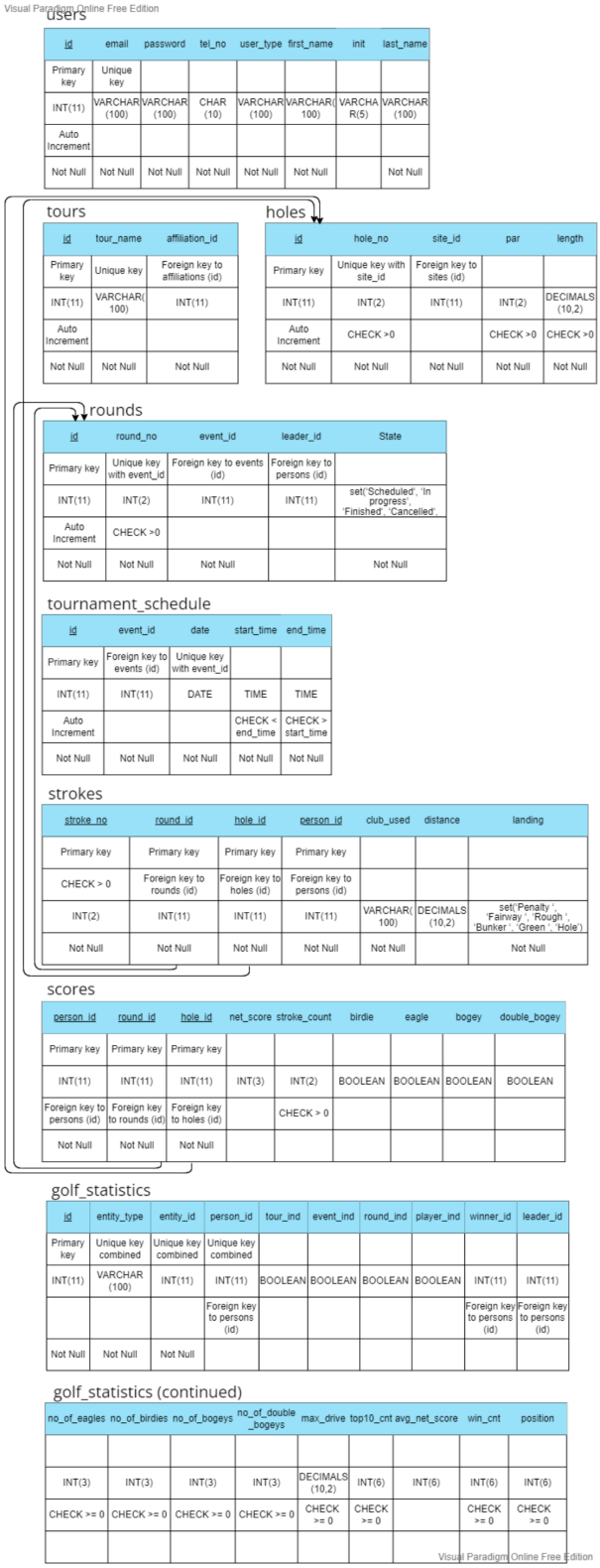
- Create new table STROKES:

```
CREATE TABLE ‘strokes‘ (
    ‘stroke_no‘ int(2) NOT NULL,
    ‘round_id‘ int(11) NOT NULL,
    ‘hole_id‘ int(11) NOT NULL,
    ‘person_id‘ int(11) NOT NULL,
    ‘club_used‘ varchar(100) NOT NULL,
    ‘distance‘ decimal(10,2) DEFAULT NULL,
    ‘landing‘ set('Penalty','Fairway','Rough','Bunker','Green','Hole') NOT NULL,
    PRIMARY KEY (‘stroke_no‘,‘round_id‘,‘hole_id‘,‘person_id‘),
    KEY ‘FK_strokes_person_id__persons_id‘ (‘person_id‘),
    KEY ‘FK_strokes_round_id__rounds_id‘ (‘round_id‘),
    KEY ‘FK_strokes_hole_id__holes_id‘ (‘hole_id‘),
    CONSTRAINT ‘FK_strokes_hole_id__holes_id‘ FOREIGN KEY (‘hole_id‘) REFERENCES
     ‘holes‘ (‘id‘),
    CONSTRAINT ‘FK_strokes_person_id__persons_id‘ FOREIGN KEY (‘person_id‘) REFERENCES
     ‘persons‘ (‘id‘),
    CONSTRAINT ‘FK_strokes_round_id__rounds_id‘ FOREIGN KEY (‘round_id‘) REFERENCES
     ‘rounds‘ (‘id‘),
    CONSTRAINT ‘CHK_stroke_no‘ CHECK (‘stroke_no‘ > 0)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

- Create new table GOLF_STATISTICS:

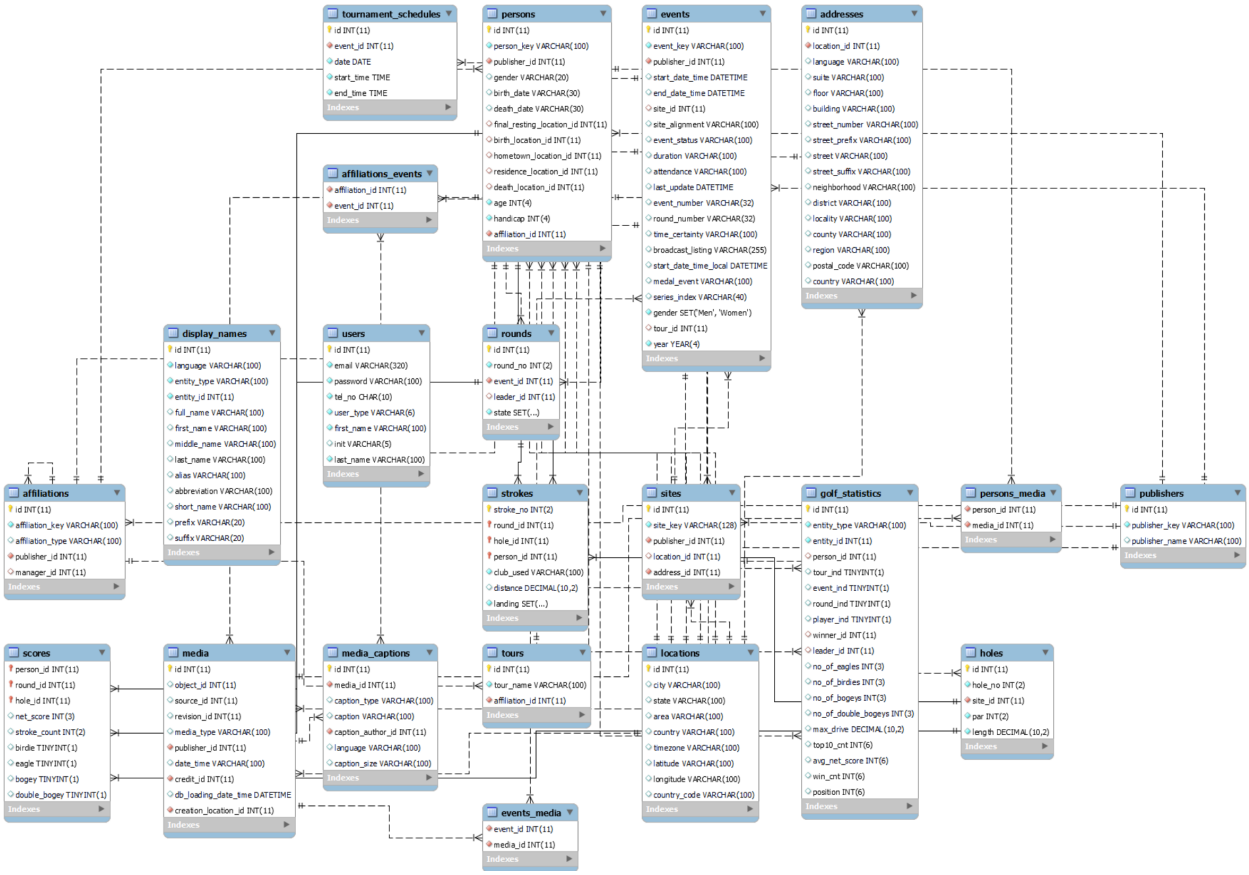
```
CREATE TABLE ‘golf_statistics‘ (
    ‘id‘ int(11) NOT NULL,
    ‘entity_type‘ varchar(100) NOT NULL,
    ‘entity_id‘ int(11) NOT NULL,
    ‘person_id‘ int(11) DEFAULT NULL,
    ‘tour_ind‘ tinyint(1) DEFAULT NULL,
    ‘event_ind‘ tinyint(1) DEFAULT NULL,
    ‘round_ind‘ tinyint(1) DEFAULT NULL,
    ‘player_ind‘ tinyint(1) DEFAULT NULL,
    ‘winner_id‘ int(11) DEFAULT NULL,
    ‘leader_id‘ int(11) DEFAULT NULL,
    ‘no_of_eagles‘ int(3) DEFAULT NULL,
    ‘no_of_birdies‘ int(3) DEFAULT NULL,
    ‘no_of_bogeys‘ int(3) DEFAULT NULL,
    ‘no_of_double_bogeys‘ int(3) DEFAULT NULL,
    ‘max_drive‘ decimal(10,2) DEFAULT NULL,
    ‘top10_cnt‘ int(6) DEFAULT NULL,
    ‘avg_net_score‘ int(6) DEFAULT NULL,
    ‘win_cnt‘ int(6) DEFAULT NULL,
    ‘position‘ int(6) DEFAULT NULL,
    PRIMARY KEY (‘id‘),
    UNIQUE KEY ‘entity_type‘ (‘entity_type‘,‘entity_id‘,‘person_id‘),
    KEY ‘FK_golf_statistics_person_id__persons_id‘ (‘person_id‘),
    KEY ‘FK_golf_statistics_winner_id__persons_id‘ (‘winner_id‘),
    KEY ‘FK_golf_statistics_leader_id__persons_id‘ (‘leader_id‘),
    CONSTRAINT ‘FK_golf_statistics_leader_id__persons_id‘ FOREIGN KEY (‘leader_id‘)
     REFERENCES ‘persons‘ (‘id‘),
    CONSTRAINT ‘FK_golf_statistics_person_id__persons_id‘ FOREIGN KEY (‘person_id‘)
     REFERENCES ‘persons‘ (‘id‘),
    CONSTRAINT ‘FK_golf_statistics_winner_id__persons_id‘ FOREIGN KEY (‘winner_id‘)
     REFERENCES ‘persons‘ (‘id‘),
    CONSTRAINT ‘CHK_no_of_eagles‘ CHECK (‘no_of_eagles‘ >= 0),
    CONSTRAINT ‘CHK_no_of_birdies‘ CHECK (‘no_of_birdies‘ >= 0),
    CONSTRAINT ‘CHK_no_of_bogeys‘ CHECK (‘no_of_bogeys‘ >= 0),
    CONSTRAINT ‘CHK_no_of_double_bogeys‘ CHECK (‘no_of_double_bogeys‘ >= 0),
    CONSTRAINT ‘CHK_max_drive‘ CHECK (‘max_drive‘ >= 0),
    CONSTRAINT ‘CHK_top10_cnt‘ CHECK (‘top10_cnt‘ >= 0),
    CONSTRAINT ‘CHK_win_cnt‘ CHECK (‘win_cnt‘ >= 0),
    CONSTRAINT ‘CHK_position‘ CHECK (‘position‘ >= 0)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

4.3 Visual Diagram



4.4 Final MySQL-generated EER-Diagram

This is the final EER diagram produced by MySQL Workbench of all the tables that will be used in our project:



5 Task 5: Web Management

Our team created web pages for the following tasks:

- Manage Users
- Login
- Manage locations
- Manage addresses
- Manage courses (sites)
- Manage holes at courses
- Manage organizations
- Manage tours
- Manage tournaments
- Manage rounds
- Manage tournament schedule
- Manage players
- Register players for tournaments
- Manage scores
- Manage media
- Display statistics

Please see our GitHub repository for the source code of these pages.

6 Task 6: Sample Data

Please see the following files on our GitHub repository:

- PA5_Base_Relations.sql
- pa5_golf_schema.sql
- Sample Data folder contains several JSON files of sample data that we curated by hand.

6.1 Explanation Of Methods

For some relations, it was feasible to manually create JSON-objects within javascript (see Sample Data folder on GitHub) files and query an API that Wian created in order to populate the tables. Although this approach was convenient, it did not scale well to relations such as SCORES which required 3600+ tuples (ten players, five tournaments, four rounds per tournament and 18 holes played per round). For this relation, and others requiring a similiar number of tuples, we opted for an algorithmic approach that populated these relations with coherent data, across many tuples using nested for-loops (for an example see Sample Data/ populateScores.js). This method was chosen, firstly, because it was far more efficient than entering by hand, and, secondly, because it was orders-of-magnitude less prone to erroneous data-entry.

We also aimed to bring our sample data closer to reality by randomising outcomes of play. For instance, we randomised the number of shots it took a player, during a hole, to sink the ball in the hole and used a logarithmic progression to incorporate the likelihood of hitting a hole-in-one, 1-under par, 2-under par, etc. (please see line 31, Sample Data/ populateStrokes.js on the GitHub repository). Another example of this approach can be found on line 40 of the same file, where we rudimentarily randomised the quality of the player's shot, i.e. how well positioned the shot was (for instance, a fairway is better than a bunker).

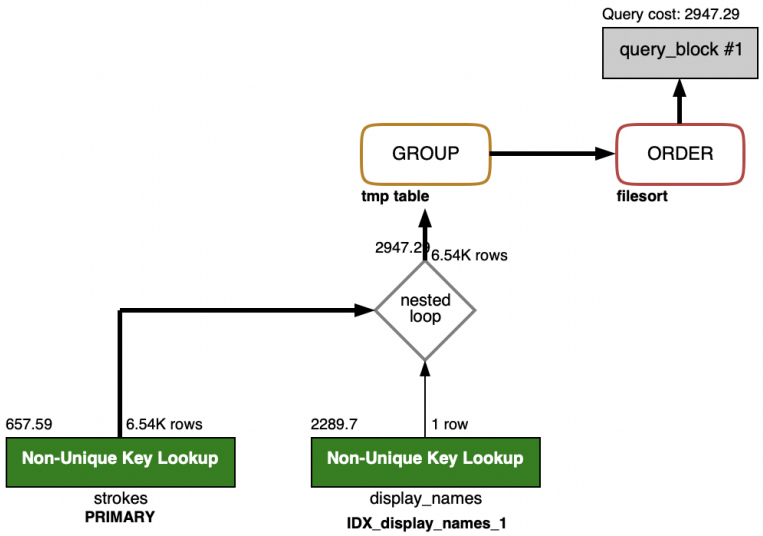
7 Task 7: Analysis and Optimisation

7.1 Query 1: Driving-Distance Leaderboard

This query gathers the top-10 players by driving distance off the tee, that is, distance achieved, on average, on the first shot of every hole.

```
SELECT first_name, last_name,
AVG(distance) as avg_tee_off
FROM strokes
LEFT JOIN display_names
ON person_id = entity_id
WHERE stroke_no = 1
GROUP BY first_name
ORDER BY avg_tee_off DESC;
```

7.1.1 Initial Analysis of Performance



For this query's initial performance metrics, we see:

- Query Cost: 2947.29
- Execution time: 0.02 seconds

7.1.2 Plan For Optimisation

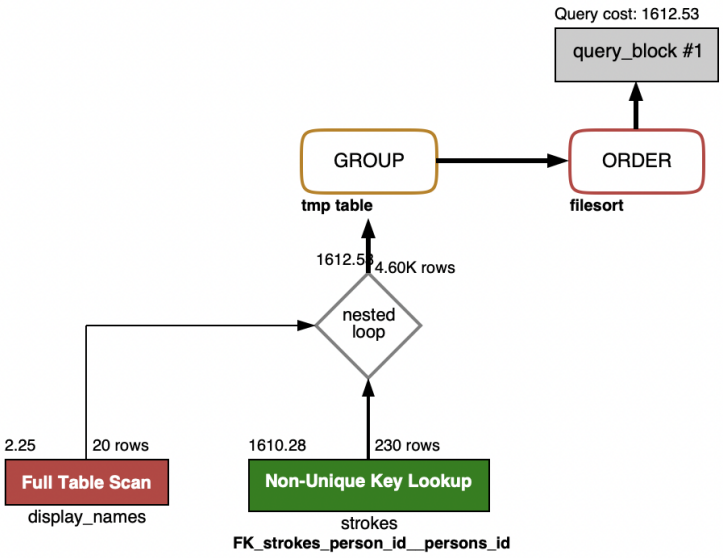
After reading some research, we found that we could immediately improve the performance of this query, while maintaining the same output, by changing the LEFT JOIN to an INNER JOIN. The effects of the optimisation are analysed below.

```
SELECT first_name, last_name,
AVG(distance) as avg_tee_off
FROM strokes
INNER JOIN display_names
ON person_id = entity_id
WHERE stroke_no = 1
GROUP BY first_name
ORDER BY avg_tee_off DESC;
```

7.1.3 Results

For the optimised query's performance metrics, we see:

- Query Cost: 1612.53
- Execution time: 0.014 seconds



With the optimisation implemented, we see a substantial improvement in the performance metrics.

7.1.4 Why Did The Optimisation Work?

LEFT JOIN returns all rows in the left table regardless of whether there is a matching value in the right table causing a sizeable decrease in performance. Since we didn't have any NULL valued attributes present in the output of the initial query, this went unnoticed.

INNER JOIN saves time by excluding any tuples with no matching attributes, thus, saving on time and resources used.

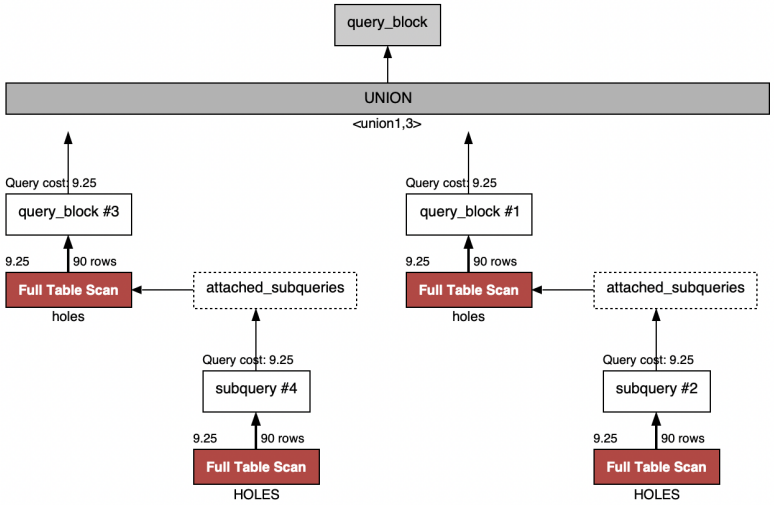
7.2 Query 2: Retrieving Shortests and Longest Holes In Database

This query, here in its initial form, retrieves the longest hole, as well as the shortest hole, in the HOLES relation by yards using two subqueries and a UNION operation.

```
SELECT id, length
FROM holes
WHERE length = (SELECT MAX(length) FROM holes)
UNION
FROM holes
WHERE length = (SELECT MIN(length) FROM holes)
```

7.2.1 Initial Analysis of Performance

- Query Cost: 37
- Execution time: 0.0012 seconds (varies by ± 0.0001 of a second)



7.2.2 Plan For Optimisation

To optimise this query, we substituted the UNION operation for an OR clause, to see if any performance gains could be made. After some research, we also decided to replace the HAVING clause with a WHERE clause.

```
SELECT id, length,
FROM holes
WHERE length = (SELECT MAX(length) FROM holes)
OR length = (SELECT MIN(length) FROM holes);
```

7.2.3 Results

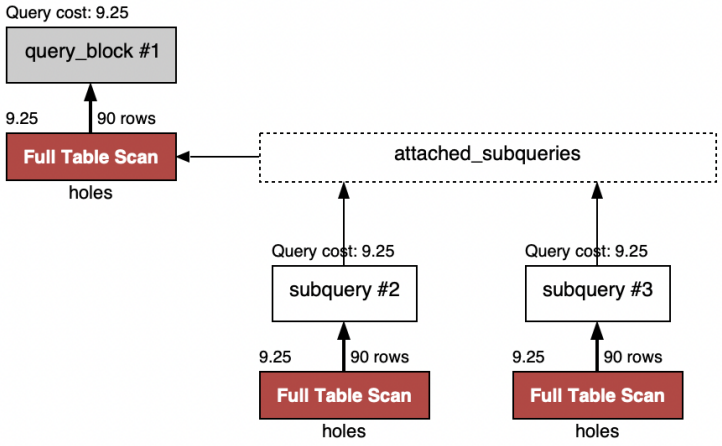
For the optimised query's performance metrics, we see:

- Query Cost: 28
- Execution time: 0.0009 seconds (varies by ± 0.0001 of a second)

With the optimisation implemented, we see clear evidence of a resource optimisation having occurred as well as potential evidence of a decrease in execution time.

7.2.4 Why Did The Optimisation Work?

From the following execution plan of the optimised query, it is clear to see why there was a 25% reduction in the number of resources used to execute the query: we no longer duplicate the work.



The WHERE clauses optimises a query because WHERE restricts the result set before retrieving rows while a HAVING clause restricts the result set after retrieving all the tuples in a relation.

8 Task 8: Comments on Development

8.1 Usage of git

We have multiple git commits, from multiple members of the team across several javascript, PHP, CSS and SQL files.

8.2 Usage of a package manager

We used the Composer package manager. (please see composer.json on GitHub)

8.3 Data validation

Multiple instances of data validation using Javascript can be found in our GitHub repository.

8.3.1 js/addresses.js

When a user wants to add or modify existing data in the addresses relation, our system first verifies that the input-language is a valid language and in the correct format. The same is done with the inputted street data and the inputted country.

8.3.2 js/locations.js

Here our system queries an API to verify that the inputted location is valid.

8.3.3 js/viewCourses.js

Here we used an in-house API to validate that the passed-in publisher, location and address are valid.

8.3.4 js/inputValidationLogin.js

Here we used ReGex to validate the name, surname, password and email required parameters.

8.3.5 js/inputValidationPlayers.js

Here we built javascript functions to validate that the user is over the age of 18.

8.3.6 js/inputValidationSignUp.js

Here, again, we used ReGex, this time to validate the name, surname, password, email, telephone number, and the initials requirement.

9 Task 9: Demo And Contributions

9.1 Contributions

u21446271 – Jonelle Coertze

- I am the manager of the team. As such, I ensured that every team member knew what was expected of them. I also followed up regularly and assisted the team in all aspects of the project. I also communicated with our mentor, Maryam if we had any questions.
- For task 1, I collected all the research done by the team members and consolidated it into the final product of task 1 as seen in this document.
- For task 2, I created the EER diagram as a draft at first, which was then used to have meetings internally and with our mentor. I also produced the final EER diagram as seen in this document.
- For task 3, I took the relational mapping that Reuben and Arno produced and added the steps as seen in this document.
- For task 4, I took the relational mapping and the SportsDB document produced by Wian and I did the final design of the golf_sportsdb. I produced the Excel mapping, the visual diagram and the SQL statements used to populate the database as seen in this document.
- For task 5, I programmed the following web pages:
  - Manage organizations
  - Manage tours
- For task 6, I produced a document which explained what needed to be done on each web page, how it maps to the database and included sample data.
- For task 8, I committed to GitHub and I ensured that I have validation in my code. I also created the README.txt document.
- For task 9, I collected the contributions from every team member and put it together as seen in this document.

u21457451 – Arno Jooste

- I was responsible for designing the web home page.
- I helped with the relational mapping.
- For task 5, I programmed the following web pages:
  - Manage tournaments
  - Manage tournament schedules page
  - Manage players' scores.
  - Register a player for a tournament,
  - Display statistics

- I also designed and implemented the sidebar menu on each page, as part of the navigation from one page to another.
- As a final contribution, I also designed the website's logo which is visible in the top left corner of each page.

u21457060 – Reuben Jooste

- I contributed a small part to task 1 which is the research part of this project. Providing the other team members with a couple of references to websites about golf and all the rules and techniques helped them quite a lot with task 1.
- I came up with the idea that the sport, Golf, would be a great choice for this project.
- I also made a big contribution to the relational mapping of the EER diagram.
- Most members contributed more to the other tasks of this practical therefore I felt like I needed to contribute more to the web programming.
- I came up with the design of the website
- For task 5, I programmed the following pages:
  - Login
  - Signup
  - Manage users
  - Manage players
  - Manage media
  - Display statistics
- I also developed the unique ways to update, create and delete the different records in the database tables. For example, instead of being directed to a new page there will be a popup asking for the input from the user.
- And finally. I provided assistance to any of the group members when they needed help with their part of the web programming.
- Overall. I feel like I did my part and fully contributed the 16.67% that each group member need to contribute (there are 6 members).

u19043512 – Wian Koekemoer

- I took an outdated sportsDB dump and updated it to match the sportsDB 29 standard.
- I wrote the documentation for the sportsDB schema that was used to reach the final iteration of our conceptual model.
- I updated the schema to match our model.
- I worked extensively to create sound data for the database, producing multiple automated files.

- For task 5, I wrote the code for the following pages:
  - Manage addresses
  - Manage locations
  - Manage courses

u19234806 – Themba Mdluli

- I did most of the work for task 1 where I wrote the foundation for the golf research.
- For task 2, I created an EER diagram that we incorporated into the final EER diagram.
- For task 5, I did the programming of the following web pages:
  - Manage rounds
  - Manage holes

u04929552 - Jake Weatherhead

- I produced the work for Task 7.
- I used my previous experience with git and GitHub to coordinate the team around best practices with regards to cloning, pulling, staging commits, forking repositories, and merging branches.
- I worked on understanding and finalising the conceptual design of the database.
- I collated the final pdf submission working, continuously, to understand, present, and explain the work produced by the group in a way that showcases our efforts.
- I have worked extensively on producing sample data, both by hand and by building an automated javascript data populator.
- Elsewhere, I volunteered whenever and wherever teammates needed my help.

10 Bibliography

"PGA Championship." Wikipedia, Wikimedia Foundation, 24 May 2022, https://en.wikipedia.orgwikiPGA_Championship

Kelley, Brent. "Here's What Counts (and Doesn't Count) as a Stroke in Golf." LiveAbout, LiveAbout, 3 Jan. 2019, https://www.liveabout.com/what-is-a-stroke-definition-1561404.

"2022 PGA Championship Leaderboard." PGA Championship, https://www.pgachampionship.com/leaderboard.

"IGF National Members." International Golf Federation, https://www.igfgolf.org/about-igf/nationalmembers/

Pgatour.com. "Golf Stat and Records: PGA Tour." PGATour, https://www.pgatour.com/stats.html