# CS303: DataBases and Information System

Suyash Gaurav 210010054

Theory Assignment 1

## 1 Q1

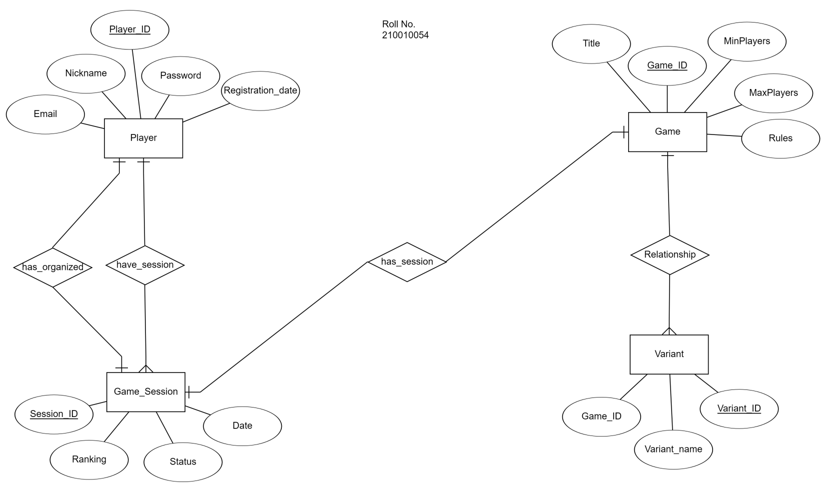


Figure 1: ER Diagram

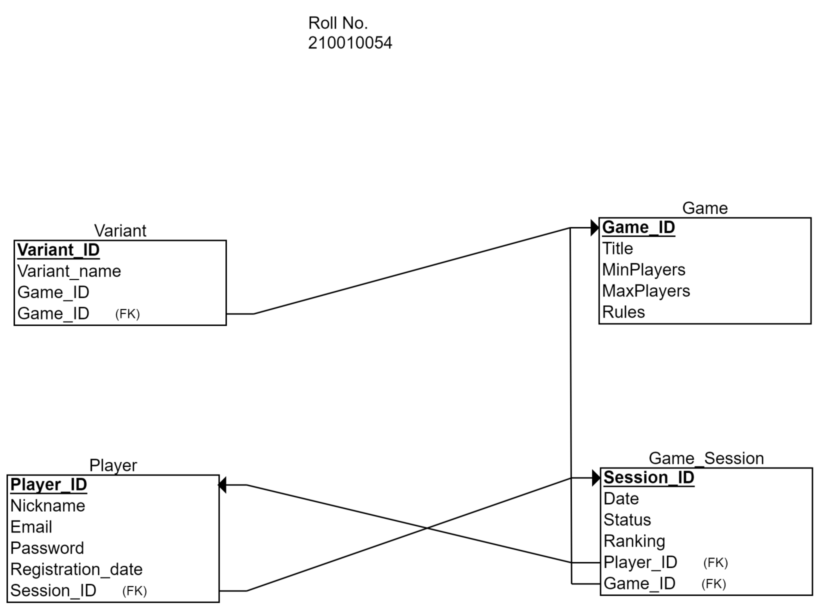


Figure 2: Relational Schema

# 2 Q2

| Activity | | |
|---|---|---|
| a_id | a_name | category |
| 1 | Judo | Combat |
| 2 | Karate | Combat |
| 3 | Martial Arts | Combat |
| 4 | Taekwondo | Combat |
| 5 | English Boxing | Boxing |
| 6 | French Boxing | Boxing |
| 7 | Cardio Boxing | Boxing |
| 8 | Diving | Aquatic sport |
| 9 | Swimming | Aquatic sport |
| 10 | Snorkeling | Aquatic sport |

| Instructor | |
|---|---|
| i_id | i_name |
| 1 | Macron |
| 2 | Trudeau |
| 3 | Merkel |
| 4 | Jacinda |
| 5 | Theresea |

| student_activity | |
|---|---|
| a_id | s_id |
| 1 | 2 |
| 1 | 1 |
| 1 | 6 |
| 1 | 8 |
| 2 | 1 |
| 2 | 2 |
| 3 | 3 |
| 3 | 5 |
| 4 | 12 |
| 4 | 11 |
| 4 | 4 |
| 5 | 4 |
| 5 | 5 |
| 5 | 6 |
| 6 | 4 |
| 7 | 7 |
| 7 | 12 |
| 8 | 9 |
| 8 | 11 |
| 9 | 6 |
| 10 | 5 |
| 10 | 4 |
| 10 | 6 |

| Student | | |
|---|---|---|
| s_id | s_name | Level |
| 1 | Stan | L3 |
| 2 | Eminem | L3 |
| 3 | Marshall | L3 |
| 4 | Marie | L2 |
| 5 | Malala | L2 |
| 6 | Diana | L2 |
| 7 | Alex | L1 |
| 8 | Peter | L1 |
| 9 | Stewie | L1 |
| 10 | Brayant | L1 |
| 11 | Rick | L1 |
| 12 | Morty | L1 |

| Instructor_activity | |
|---|---|
| a_id | i_id |
| 1 | 2 |
| 2 | 3 |
| 3 | 4 |
| 4 | 3 |
| 5 | 5 |
| 6 | 4 |
| 9 | 2 |
| 10 | 3 |

Figure 1: Database instance for Question 2.

Figure 3: Data for Q2

## 2.1 (a)

### 2.1.1 (i) Relational Algebra

$$\text{R1} \longleftarrow \Pi_{i\_id}(\sigma(instructor\_activity))$$
$$R2 \longleftarrow \Pi_{i\_id, i\_name}(\sigma_{i\_id\ not\ in\ (R1)}(instructor))$$

### 2.1.2 (ii) Tuple Relational Calculus Formula:

$$\{(t|\exists p \in Instructor(p[i\_id] = t[i\_id]) \land p[i\_name] = t[i\_name]) \land \neg(\exists q \in instructor\_activity(p[i\_id] = q[i\_id])\}$$

### 2.1.3 (iii) Domain Relational Calculus Formula:

$$\{ <\text{i\_id, i\_name} > \mid \exists \text{ a\_id (Instructor(i\_id, i\_name))} \land \neg \exists \text{ a\_id (Instructor\_activity(a\_id, i\_id))} \}$$

#### 2.1.4 (iv) Output Table

| i_id | i_name |
| --- | --- |
| 1 | Macron |

## 2.2 (b)

#### 2.2.1 (i) Relational Algebra

R1 $\longleftarrow \Pi_{i\_id}(\sigma_{category='Combat'}(Activity))$
$R2 \longleftarrow \Pi_{i\_id}(\sigma_{category='Boxing'}(Activity))$
$R3 \longleftarrow \Pi_{i\_id}(\sigma_{a\_idin(R1) \wedge a\_idin(R2)}(Instructor\_activity))$
$R4 \longleftarrow \Pi_{i\_name}(\sigma_{i\_idin(R3)}(Instructor))$

#### 2.2.2 (ii) Tuple Relational Calculus Formula:

$\{(t|\exists p \in Instrcutor(p[i\_name] = t[i\_name]) \wedge (\exists q \in Instructor\_activity(p[i\_id] = q[i\_id]) \wedge (\exists r \in Activity(r[s\_id] = q[s\_id]) \wedge r[category] =' Combat'))$
$\wedge(\exists p1 \in Instrcutor(p1[i\_name] = t[i\_name]) \wedge (\exists q1 \in Instructor\_activity(p1[i\_id] = q1[i\_id]) \wedge (\exists r1 \in Activity(r1[s\_id] = q1[s\_id]) \wedge r1[category] =' Boxing'))\}$

#### 2.2.3 (iii) Domain Relational Calculus Formula:

$\{(\text{name})\ |\exists id(Instructor(id, name) \wedge \exists a\_id, i\_id(Instructor\_activity(a\_id, i\_id) \wedge$
$Activity(a\_id, category) \wedge$
$category =' Boxing') \wedge \exists a\_id', i\_id'$
$(Instructor\_activity(a\_id', i\_id') \wedge Activity(a\_id', category') \wedge category' ='$
$Combat'))\}$

#### 2.2.4 (iv) Output Table

| i_name |
| --- |
| Jacinda |

## 2.3 (c)

#### 2.3.1 (i) Relational Algebra

R1 $\longleftarrow \Pi_{s\_id}(\sigma_{Level='L2'}(Student))$
$R2 \longleftarrow \Pi_{a\_id}(\sigma_{s\_id\ not\ in\ (R1)}(student\_activity))$
$R3 \longleftarrow \Pi_{a\_name}(\sigma_{a\_id\ in\ (R2)}(Activity))$

#### 2.3.2 (ii) Tuple Relational Calculus Formula:

$$\{(t|\exists s \in Activity(s[a\_name] = t[a\_name])\neg(\exists p \in Activity(p[a\_name] = t[a\_name])\wedge(\exists q \in student\_activity(p[a\_id] = q[a\_id])\wedge(\exists r \in student(r[s\_id] = q[s\_id]) \wedge r[Level] =' L2')))\}$$

#### 2.3.3 (iii) Domain Relational Calculus Formula:

$$\{(\text{a\_name}) \,|\forall a\_id(Activity(a\_id, a\_name) \rightarrow \neg\exists s\_id(Student\_activity(a\_id, s\_id)\wedge \exists level(Student(s\_id, level) \wedge level =' L2'))\}$$

#### 2.3.4 (iv) Output Table

| a_name |
|---|
| Karate |
| Cardio Boxing |
| Diving |

### 2.4 (d)

#### 2.4.1 (i) Relational Algebra

$$\text{R1} \longleftarrow \Pi_{s\_id}(\sigma_{Level='L2'}(Student))$$
$$R2 \longleftarrow \Pi_{a\_id}(\sigma_{s\_id\ in\ ALL(R1)}(student\_activity))$$
$$R3 \longleftarrow \Pi_{i\_id}(\sigma_{a\_id\ in\ (R2)}(Instructor\_activity))$$
$$R4 \longleftarrow \Pi_{i\_name}(\sigma_{i\_id\ in\ (R3)}(Instructor))$$

#### 2.4.2 (ii) Tuple Relational Calculus Formula:

$$\{(t|\exists p \in Instructor(p[i\_iname] = t[i\_iname])\wedge(\exists q \in instructor\_activity(p[i\_id] = q[i\_id])\wedge(\exists r \in student\_activity(r[s\_id] = q[s\_id])\wedge(\exists l \in student(l[s\_id] = r[s\_id]) \wedge l[level] =' L2'))\}$$

#### 2.4.3 (iii) Domain Relational Calculus Formula:

$$\{(\text{i\_name}) \,|\forall i\_id, a\_id(Instructor(i\_id, i\_name)\wedge Instructor\_activity(i\_id, a\_id) \rightarrow \neg\exists s\_id(Student(s\_id, level)\wedge level =' L2'\wedge\exists a\_id'(Student\_activity(s\_id, a\_id')\wedge a\_id' = a\_id))\}$$

#### 2.4.4 (iv) Output Table

| i_name |
|---|
| Merkel |
| Theresea |

## 2.5 (e)

#### 2.5.1 (i) Relational Algebra

$$\mathrm{R1} \longleftarrow \Pi_{s\_id}(\sigma_{Level='L1'}(Student))$$
$$R2 \longleftarrow \Pi_{s\_id}(\sigma_{Level='L2'}(Student))$$
$$R3 \longleftarrow \Pi_{a\_id}(\sigma_{s\_id\ in\ any(R1) \wedge s\_id\ in\ any(R2)}(student\_activity))$$
$$R4 \longleftarrow \Pi_{category}(\sigma_{a\_id\ in\ (R3)}(Activity))$$

#### 2.5.2 (ii) Tuple Relational Calculus Formula:

$$\{(t | \exists p \in activity(p[category] = t[category]) \wedge (\exists q \in student\_activity(p[a\_id] = q[a\_id]) \wedge (\exists r \in student(r[s\_id] = q[s\_id]) \wedge r[level] =' L2')) \wedge (\exists p1 \in activity(p1[category] = t[category]) \wedge (\exists q1 \in student\_activity(p1[a\_id] = q1[a\_id]) \wedge (\exists r1 \in student(r1[s\_id] = q1[s\_id]) \wedge r1[level] =' L1'))\}$$

#### 2.5.3 (iii) Domain Relational Calculus Formula:

$$\{(\text{category}) \text{ — } \forall a\_id(Activity(a\_id, category) \rightarrow (\exists s\_id(Student\_activity(a\_id, s\_id) \wedge \exists level(Student(s\_id, level) \wedge level =' L1')) \wedge (\exists s\_id'(Student\_activity(a\_id, s\_id') \wedge \exists level'(Student(s\_id', level') \wedge level' =' L2')))\}$$

#### 2.5.4 (iv) Output Table

| Category |
|---|
| Combat |

# 3 Q3

**Given Table R:**

| A1 | A2 | A3 |
|---|---|---|
| a | p | 1 |
| b | p | 2 |
| a | q | 1 |
| c | p | 4 |
| d | r | 1 |

### 3.1 (a) {A3} → {A1}

**No**,
Because same value of A3 is giving different value of A1 for different tuples. we can see that A3 = 1 is associated with both A1 = a and A1 = d. Therefore, A3 does not functionally determine A1. This functional dependency does not exist.

$$t_1[A_3] = t_5[A_3] \ \mathit{but} \ t_1[A_1] \neq t_5[A_1]$$

### 3.2 (b) {A1} → {A3}

**Yes**

### 3.3 (c) {A2} → {A3}

**No**,
because same value of A2 is giving different value of A3 for different tuples.

$$t_1[A_2] = t_2[A_2] \ \mathit{but} \ t_1[A_3] \neq t_2[A_3]$$

### 3.4 (d) {A1,A2} → {A1,A2}

**Yes**
this functional dependency essentially says that A1 and A2 determine themselves, which is trivially true. So, the functional dependency A1, A2 → A1, A2 holds.

### 3.5 (e) {A1,A3} → {A3}

**Yes**,
if we know A1 = 'a' and A2 = 'p', A3 is 1. If we know A1 = 'b' and A2 = 'p', A3 is 2. Therefore, A1, A2 uniquely determines A3. The correct answer is "Yes."

## 4 Q4

To determine whether the given schema is in BCNF (Boyce-Codd Normal Form) and 3NF (Third Normal Form), we need to analyze the functional dependencies and identify any violations.

**Given Functional Dependencies:**
**FD1: B1, B2 → B4**
**FD2: B2, B3 → B5**
**FD3: B5 → B6**

### 4.1 (i) BCNF (Boyce-Codd Normal Form)

Condition to be satisfied:
**For every non-trivial functional dependency X → Y in the set of functional dependencies F, X must be a superkey.**

The attributes B1, B2, B3 are a candidate key since they form the primary key.

Now, let's check if FD1, FD2, and FD3 violate the BCNF requirements:

(i) FD1: B1, B2 → B4 B1, B2 is a superkey (it contains a candidate key), so FD1 satisfies BCNF.

(ii) FD2: B2, B3 → B5 B2, B3 is a superkey (it contains a candidate key), so FD2 satisfies BCNF.

(iii) FD3: B5 → B6 B5 is not a superkey because it doesn't contain a candidate key. Therefore, FD3 violates BCNF.

Since FD3 violates BCNF, we need to decompose the schema to achieve BCNF. To do this, we can create a new schema where each relation has a superkey on the left-hand side of its functional dependencies. In this case, we can create two tables:

(i) R1(B5, B6) with FD3: B5 → B6
(ii) R2(B1, B2, B4) with FD1: B1, B2 → B4

### 4.2 (ii) 3NF (Third Normal Form)

**Condition to be satisfied:**
**(i) The schema must be in BCNF.**
**(ii) For every non-trivial functional dependency X → Y in F, X must be a superkey or X must be a subset of a superkey.**

Since we've already ensured that the schema is in BCNF, let's check if FD1 and FD2 satisfy the second requirement:

(i) FD1: B1, B2 → B4
(ii) B1, B2 is a superkey, so FD1 satisfies 3NF. FD2: B2, B3 → B5

B2, B3 is not a superkey, but it is a subset of a superkey B1, B2, B3.

Therefore, FD2 satisfies 3NF. The schema satisfies both BCNF and 3NF requirements after decomposition, and there are no further violations.