Legal Management System
Group 1

Normalization and Schema Refinement

MC212: Database Management Systems

Full Name	Enrollment No.
Vedanshee Patel	202203003
Preksha Shah	202203004
Rohan Jasani	202203007
Aum Vadodaria	202203017
Madhav Kanjilimadom	202203018
Vyom Narsana	202203026

Course Instructor: Prof. Rachit Chhaya

Department of Mathematics and Computing
Dhirubhai Ambani Institute of Information and Communication Technology
Gandhinagar, Gujarat, India

October 20, 2023

Contents

1 | ER Diagram

We have made some more changes in the ER Diagram. We have attached ER Diagram as a separate submission in this lab itself as .png file.

2 | Normalization and Schema Refinement of Entity Sets

2.1 | Case

- Case(Case ID, Case type, Case Result, Case Status, Judge ID, Case Details, Defending Lawyer ID, Defendant ID, Accuser ID, Prosecuting lawyer ID) has 10 attributes.
  - CK = {(Case ID)}
  - PK = {(Case ID)}
- Functional Dependencies:
  - Case ID → (Case type, Case Result, Case Status, Judge ID, Case Details, Defending Lawyer ID, Defendant ID, Accuser ID, Prosecuting lawyer ID)
- To achieve 1NF, we refine the schema as follows:
  - Case(Case ID, Case Result, Case Status, Judge ID, Case Details, Defending Lawyer ID, Defendant ID, Accuser ID, Prosecuting lawyer ID)

    ∪

    Case_Type(Case ID, Case Type)
- For Case1:
  - CK = {(Case ID)}
  - PK = {(Case ID)}
- Functional Dependencies:
  - Case ID → (Case Result, Case Status, Judge ID, Case Details, Defending Lawyer ID, Defendant ID, Accuser ID, Prosecuting lawyer ID)
- No multi-valued attributes, already in 1NF.
  - No partial dependencies ⇒ 2NF
  - No transitive dependencies ⇒ 3NF
  - All attributes depend on a candidate key ⇒ BCNF
- For Case_Type:
  - CK = {(Case ID, Case Type)}
  - PK = {(Case ID, Case Type)}
- No functional dependencies, no multi-valued attributes.
  - No partial dependencies ⇒ 2NF
  - No transitive dependencies ⇒ 3NF
  - No dependencies ⇒ BCNF
- Thus, Case1 and Case_Type are both in BCNF forms.

2.2 | Lawyer Schema

- Lawyer(ID, email ID, name, Education, Area of expertise, Salary, License Year, Experience) has 8 attributes.
  - CK = {(ID), (email ID)}
  - PK = {(ID)}
- Functional Dependencies:
  - ID → (name, Education, Area of expertise, Salary, License Year, Experience)
  - email ID → (name, Education, Area of expertise, Salary, License Year, Experience)
  - ID ↔ Email ID
- To achieve 1NF, we refine the schema as follows:
  - Lawyer(ID, email ID, name, Education, Salary, License Year, Experience)

    ∪

    Lawyer_Area_of_Expertise(ID, Area of expertise)
- For Lawyer1:
  - CK = {(ID), (email ID)}
  - PK = {(ID)}
- Functional Dependencies:
  - ID → (name, Education, Salary, License Year, Experience)
  - email ID → (name, Education, Salary, License Year, Experience)
  - ID ↔ email ID
- No multi-valued attributes, already in 1NF.
  - No partial dependencies ⇒ 2NF
  - No transitive dependencies ⇒ 3NF
  - All attributes depend on a candidate key ⇒ BCNF
- For Lawyer_Area_of_Expertise:
  - CK = {(ID, Area of expertise)}
  - PK = {(ID, Area of expertise)}
- No functional dependencies, no multi-valued attributes.
  - No partial dependencies ⇒ 2NF
  - No transitive dependencies ⇒ 3NF
  - No dependencies ⇒ BCNF
- Thus, Lawyer1 and Lawyer_Area_of_Expertise are both in BCNF forms.

2.3 | Paralegals and Trainees

- Paralegals and Trainees(ID, name, Independent Lawyer ID, Firm ID, Salary, Experience, Education, Position) has 8 attributes.
  - CK = {(ID)}
  - PK = {(ID)}
- Functional Dependencies:
  - ID → (name, Independent Lawyer ID, Firm ID, Salary, Experience, Education, Position)

- To achieve 1NF, we refine the schema as follows:
  - Paralegals and Trainees(ID, name, Independent Lawyer ID, Firm ID, Salary, Experience, Education, Position)

    $\cup$

    PNT_Independent_Lawyer(ID, Independent Lawyer ID)
- For Paralegals and Trainees1:
  - CK = {(ID)}
  - PK = {(ID)}
- Functional Dependencies:
  - ID $\rightarrow$ (name, Firm ID, Salary, Experience, Education, Position)
- To further normalize the multivalued attribute "Firm ID":
  - Paralegals and Trainees1(ID, name, Salary, Experience, Education, Position)

    $\cup$

    PNT_Firm(ID, Firm ID)
- For Paralegals and Trainees2:
  - CK = {(ID)}
  - PK = {(ID)}
- Functional Dependencies:
  - ID $\rightarrow$ (name, Salary, Experience, Education, Position)
- No multivalued attributes, already in 1NF.
  - No partial dependencies $\Rightarrow$ 2NF
  - No transitive dependencies $\Rightarrow$ 3NF
  - All attributes depend on a candidate key $\Rightarrow$ BCNF
- For PNT_Independent_Lawyer and PNT_Firm:
  - Composite Candidate Key (CK) = {(ID, Independent Lawyer ID)} and {(ID, Firm ID)} respectively
  - PK = {(ID, Independent Lawyer ID)} and {(ID, Firm ID)} respectively
- No functional dependencies, no multivalued attributes.
  - No partial dependencies $\Rightarrow$ 2NF
  - No transitive dependencies $\Rightarrow$ 3NF
  - No dependencies $\Rightarrow$ BCNF
- Thus Paralegals and Trainees2, PNT_Independent_Lawyer, and PNT_Firm are in BCNF forms.

2.4 | Client

- Client(ID, Client type, name, Client Details) has 4 attributes.
  - CK = {(ID)}
  - PK = {(ID)}
- Functional Dependencies:
  - ID → (Client type, name, Client Details)
- To achieve 1NF, we refine the schema as follows:
  - Client(ID, Client type, name, Client Details)

    ∪

    Client_Type(ID, Client Type)
- For Client1:
  - CK = {(ID)}
  - PK = {(ID)}
- Functional Dependencies:
  - ID → (name, Client Details)
- No multivalued attributes, already in 1NF.
  - No partial dependencies ⇒ 2NF
  - No transitive dependencies ⇒ 3NF
  - All attributes depend on a candidate key ⇒ BCNF
- For Client_Type:
  - Composite Candidate Key (CK) = {(ID, Client Type)}
  - PK = {(ID, Client Type)}
- No functional dependencies, no multivalued attributes.
  - No partial dependencies ⇒ 2NF
  - No transitive dependencies ⇒ 3NF
  - No dependencies ⇒ BCNF
  - Thus Client1 and Client_Type are both in BCNF forms.

2.5 | Judges

- Judges(Judge ID, Name, Experience, Salary, License year, Education) have 6 attributes.
  - CK = {(Judge ID)}
  - PK = {(Judge ID)}
- Functional Dependencies:
  - Judge ID → (Name, Experience, Salary, License year, Education)
- No multivalued attributes, already in 1NF.
  - No partial dependencies ⇒ 2NF
  - No transitive dependencies ⇒ 3NF
  - All attributes depend on a candidate key ⇒ BCNF
- Thus, Judges is already in BCNF form and no further refinement is required.

2.6 | Task

- Task(Task from ID, Task Number, Task To ID, Case ID, Task Status, Task Description) has 6 attributes.
  - CK = {(Task from ID, Task Number, Task To ID)}
  - PK = {(Task from ID, Task Number, Task To ID)}
- Functional Dependencies:
  - (Task from ID, Task Number, Task To ID) → (Case ID, Task Status, Task Description)
- To achieve 1NF, we refine the schema as follows:
  - Task(Task from ID, Task Number, Task To ID, Task Status, Task Description)

    ∪

    Task_Case(Task from ID, Task Number, Task To ID, Case ID)
- For Task1:
  - CK = {(Task from ID, Task Number, Task To ID)}
  - PK = {(Task from ID, Task Number, Task To ID)}
- Functional Dependencies:
  - (Task from ID, Task Number, Task To ID) → (Task Status, Task Description)
- No multivalued attributes, already in 1NF. No partial dependencies or transitive dependencies.
- All attributes depend on the entire composite primary key ⇒ BCNF
- For Task_Case:
  - Composite Candidate Key (CK) = {(Task from ID, Task Number, Task To ID, Case ID)}
  - PK = {(Task from ID, Task Number, Task To ID, Case ID)}
- No functional dependencies, no multivalued attributes.
  - No partial dependencies ⇒ 2NF
  - No transitive dependencies ⇒ 3NF
  - No dependencies ⇒ BCNF
- Thus Task1 and Task_Case are in BCNF forms.

2.7 | Firm

- Firm(Firm ID, Firm name, Establishment year, Expenditure) has 4 attributes.
  - CK = {(Firm ID), (Firm name)}
  - PK = {(Firm ID)}
- Functional Dependencies:
  - Firm ID → (Establishment year, Expenditure)
  - Firm name → (Establishment year, Expenditure)
  - Firm ID ↔ Firm name
- No multivalued attributes, already in 1NF.
  - No partial dependencies ⇒ 2NF
  - No transitive dependencies ⇒ 3NF
  - All attributes depend on a candidate key ⇒ BCNF
- Thus, Firm is already in BCNF form with its two candidate keys.

2.8 | Bar Council

- Bar Council(ID, Name) has 2 attributes.
  - CK = {(ID)}
  - PK = {(ID)}
- Functional Dependencies:
  - ID → (Name)
- No multivalued attributes, already in 1NF.
  - No partial dependencies ⇒ 2NF
  - No transitive dependencies ⇒ 3NF
  - All attributes depend on a candidate key ⇒ BCNF
- Thus, Bar Council is already in BCNF form, and no further refinement is required.

2.9 | Client Case Requests/Case Assistance Requests Schema

- Client Case Requests/Case Assistance Requests(Request from, Request to, Case ID, Request type, Request status) has 5 attributes.
  - CK = {(Request from, Case ID, Request to)}
  - PK = {(Request from, Case ID, Request to)}
- Functional Dependencies:
  - (Request from, Case ID, Request to) → (Request type, Request status)
- No multivalued attributes, already in 1NF. No partial dependencies or transitive dependencies.
  - All attributes depend on the entire composite primary key ⇒ BCNF
- Thus, Client Case Requests/Case Assistance Requests is already in its BCNF form.

2.10 | Joining Requests Schema

- Joining Requests(Request from, Request to, Request status) has 3 attributes.
  - CK = {(Request from, Request to)}
  - PK = {(Request from, Request to)}
- Functional Dependencies:
  - (Request from, Request to) → (Request status)
- No multivalued attributes, already in 1NF. No partial dependencies or transitive dependencies.
  - All attributes depend on the entire composite primary key ⇒ BCNF
- Thus, Joining Requests is already in its BCNF form.

2.11 | General Queries Schema

- General Queries(Query from ID, Query No, Query Handler ID, Query status, Query Description) has 5 attributes.
  - CK = {(Query From ID, Query No)}
  - PK = {(Query From ID, Query No)}
- Functional Dependencies:
  - (Query From ID, Query No) → (Query Handler ID, Query Status, Query Description)
- No multivalued attributes, already in 1NF. No partial dependencies or transitive dependencies.
  - All attributes depend on the entire composite primary key ⇒ BCNF
- Thus, General Queries is already in its BCNF form.

2.12 | License Request Queries Schema

License Request Queries(Query from ID, Email ID, Query Handler ID, Query status, Name, Resume, CLAT Score) has 7 attributes.

- CK = {(Query From ID), (Email ID)}
- PK = {(Query From ID)}

Functional Dependencies:

- (Query From ID) → (Query Handler ID, Query status, Name, Resume, CLAT Score)
- (Email ID) → (Query Handler ID, Query status, Name, Resume, CLAT Score)
- (Query From ID) ↔ (Email ID)

No multivalued attributes, already in 1NF. No partial dependencies or transitive dependencies.

- All attributes depend on a candidate key ⇒ BCNF

Thus, License Request Queries is already in its BCNF form.

3 | Normalization and Schema Refinement of Relationship Sets

3.1 | Hires Schema

- Hires(Hires from ID, Hires to ID) has 2 attributes.
  - CK = {(Hires from ID, Hires to ID)}
  - PK = {(Hires from ID, Hires to ID)}
- There are no functional dependencies, as the only two attributes of Hires form a composite primary key.
- No multivalued attributes, already in 1NF. No partial dependencies or transitive dependencies.
  - No dependencies at all ⇒ BCNF
- Thus, Hires is already in its BCNF form.

3.2 | Supervises Schema

- Supervises(Supervisor ID, Supervisee ID) has 2 attributes.
  - CK = {(Supervisor ID, Supervisee ID)}
  - PK = {(Supervisor ID, Supervisee ID)}
- There are no functional dependencies, as the only two attributes of Supervises form a composite primary key.
- No multivalued attributes, already in 1NF. No partial dependencies or transitive dependencies.///»» hello aa ck and pk ma be bracket rakhvan ache ke ek j rakhvu che??
  - No dependencies at all ⇒ BCNF
- Thus, Supervises is already in its BCNF form.

3.3 | Associated Schema

- Associated(Associated to ID, Associated with ID) has 2 attributes.
  - CK = {(Associated to ID, Associated with ID)}
  - PK = {(Associated to ID, Associated with ID)}
- There are no functional dependencies, as the only two attributes of Associated form a composite primary key.
- No multivalued attributes, already in 1NF. No partial dependencies or transitive dependencies.
  - No dependencies at all ⇒ BCNF
- Thus, Associated is already in its BCNF form.

3.4 | Manages Schema

- Manages(Manager ID, Employee ID) has 2 attributes.
  - CK = {(Manager ID, Employee ID)}
  - PK = {(Manager ID, Employee ID)}
- There are no functional dependencies, as the only two attributes of Manages form a composite primary key.
- No multivalued attributes, already in 1NF. No partial dependencies or transitive dependencies.
  - No dependencies at all ⇒ BCNF
- Thus, Manages is already in its BCNF form.

4 | Final Schemas

1. Case1(Case ID, Case Result, Case Status, Judge ID, Case Details, Defending Lawyer ID, Defendant ID, Accuser ID, Prosecuting lawyer ID)
2. Case_Type(Case ID, Case Type)
3. Lawyer1(ID, email ID, name, Education, Salary, License Year, Experience)
4. Lawyer_Area_of_Expertise(ID, Area of expertise)

5. Paralegals_and_Trainees2(ID, name, Salary, Experience, Education, Position)
6. PNT_Independent_Lawyer(ID, Independent Lawyer ID)
7. PNT_Firm(ID, Firm ID)
8. Client1(ID, name, Client Details)
9. Client_Type(ID, Client Type)
10. Judges(Judge ID, Name, Experience, Salary, Licence year, Education)
11. Task1(Task from ID, Task Number, Task To ID, Task Status, Task Description)
12. Task_Case(Task from ID, Task Number, Task To ID, Case ID)
13. Firm(Firm ID, Firm name, Establishment year, Expenditure)
14. Bar_Council(ID, Name)
15. Client_Case_Requests/Case_Assistance_Requests(Request from, Request to, Case ID, Request type, Request status)
16. Joining_Request(Request from, Request to, Request status)
17. General_Queries(Query from ID, Query No, Query Handler ID, Query status, Query Description)
18. License_Request_Queries(Query from ID, Email ID, Query Handler ID, Query status, Name, Resume, CLAT Score)
19. Manages(Manager ID, Employee ID)
20. Supervises(Supervisor ID, Supervisee ID)
21. Associated(Associated to ID, Associated with ID)
22. Hires(Hires from ID, Hires to ID)