Project Title: - Insurance Agency Management System

Description

An insurance policy/plan is a contract between an individual (Policyholder) and an insurance company (Provider). Under the agreement, Individuals pay regular amounts of money (as premiums) to the insurer, and the company pays settlement if the sum assured on unfortunate event arises.

The choice of a specific type of insurance policy is made based on individual needs and life goals. Clients are sustained under agents. One client can take one or many policies.

Our insurance company mainly provide three types of policies which are:

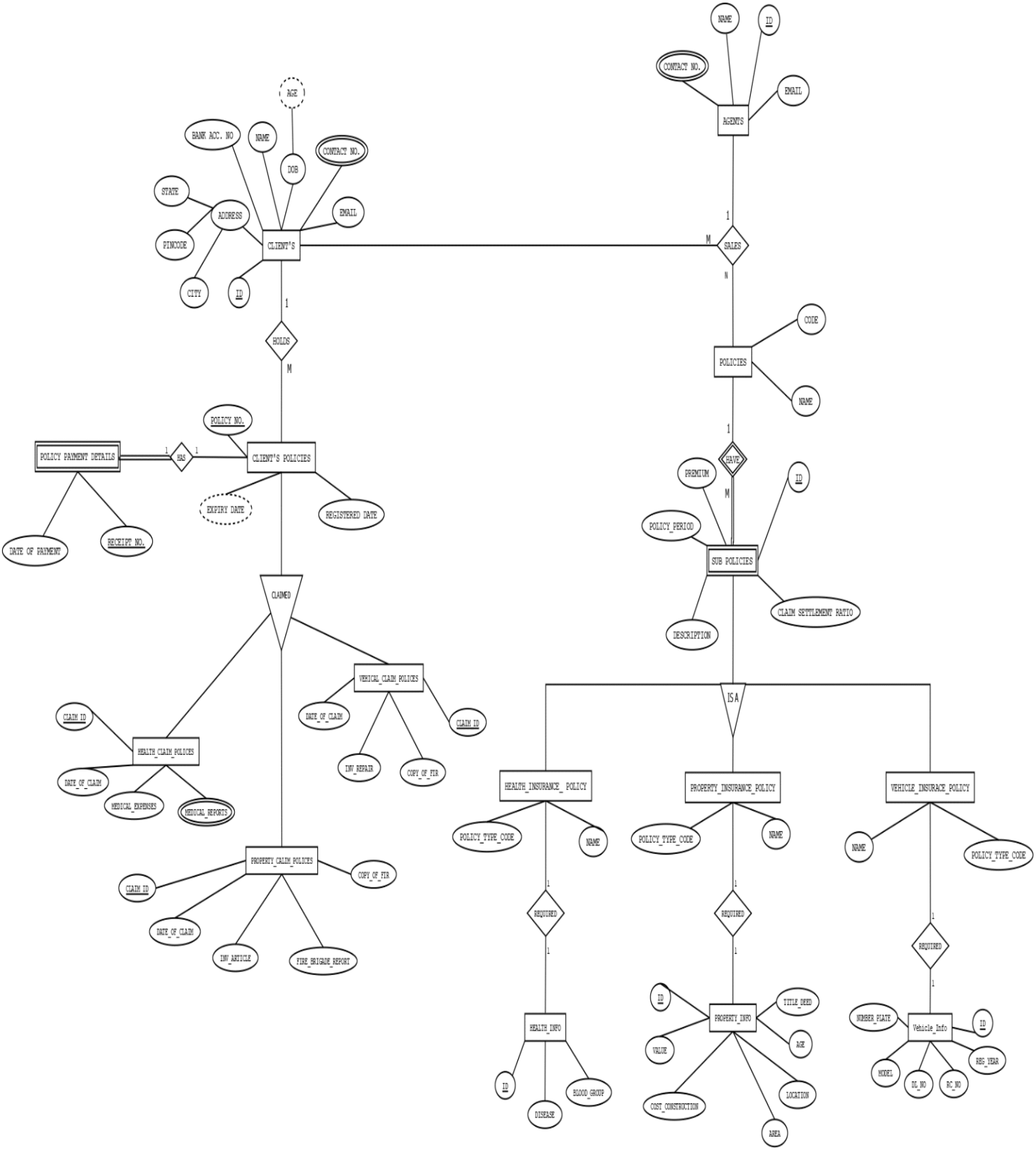
1. Health Insurance
2. Property Insurance
3. Vehicle Insurance

To purchase a policy, an individual must provide necessary documents and personal information to check the eligibility of an insured individual for the specific type of insurance policy that they want to buy.

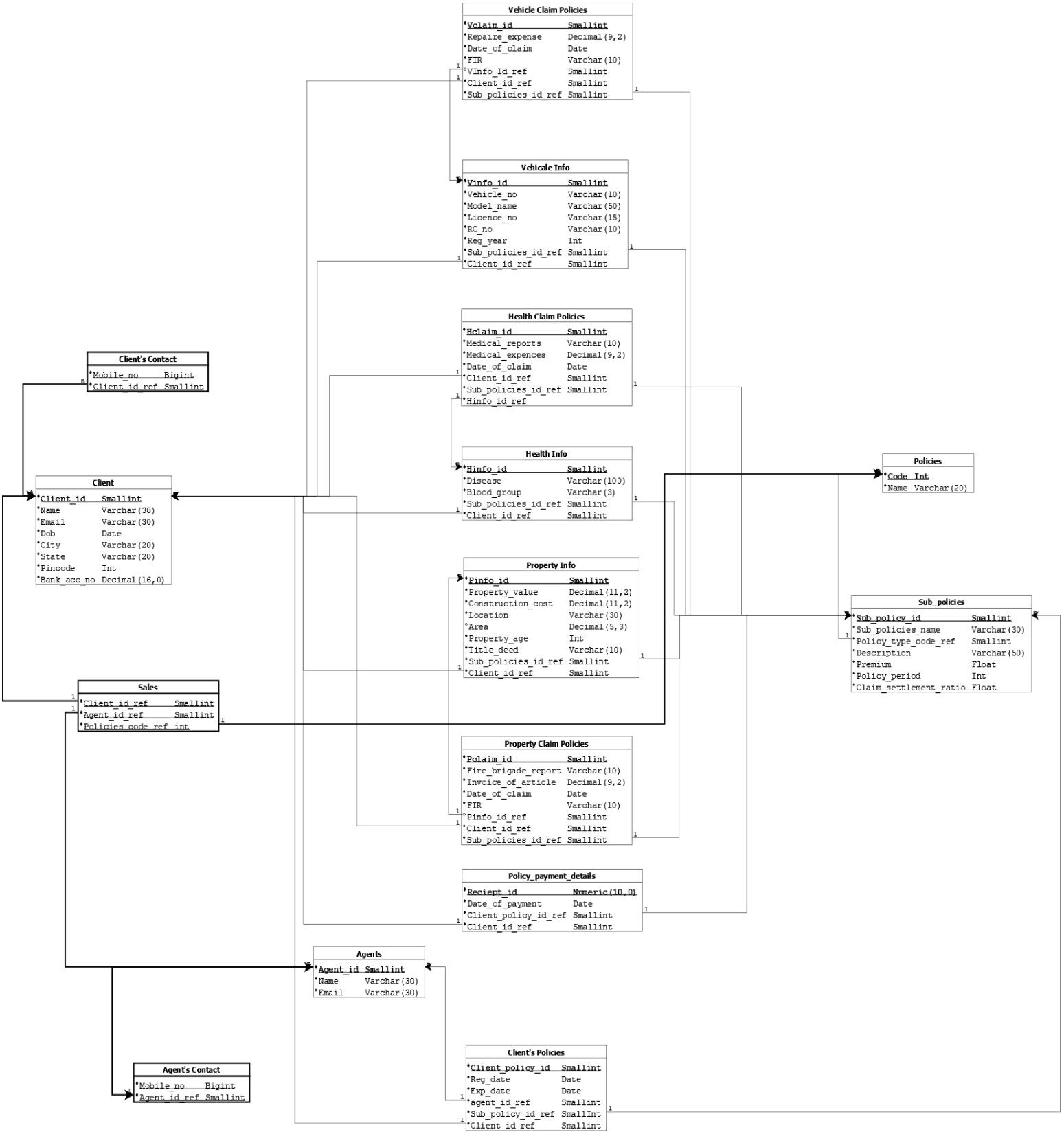
The premium of an insurance policy is the amount that individuals need to pay to purchase a specific insurance cover. It is typically expressed as a regular cost, monthly, quarterly, half-yearly, or annually. Validity of policy is determined based on the period (policy term), loss or injury, and similar other factors.

Claim settlement ratio is the percentage of claims that we settle in a year out of the total claim. It acts as an indicator of their credibility. As a general rule, the higher the ratio, the more reliable the insurer is.

ERD


NAME
ID
CONTACT NO.
EMAIL
AGENTS
AGE
BANK ACC. NO
NAME
DOB
CONTACT NO.
STATE
ADDRESS
EMAIL
PINCODE
CLIENT'S
CITY
ID
1
M
SALES
N
1
HOLDS
M
CODE
POLICIES
NAME
POLICY NO.
POLICY PAYMENT DETAILS
1
HAS
1
CLIENT'S POLICIES
1
HAVE
PREMIUM
ID
M
EXPIRY DATE
REGISTERED DATE
POLICY_PERIOD
DATE OF PAYMENT
RECEIPT NO.
SUB POLICIES
CLAIMED
CLAIM SETTLEMENT RATIO
DESCRIPTION
VEHICAL_CLAIM_POLICES
IS A
CLAIM ID
DATE_OF_CLAIM
CLAIM ID
HEALTH_CLAIM_POLICES
INV_REPAIR
COPY_OF_FIR
DATE_OF_CLAIM
MEDICAL_EXPENSES
MEDICAL_REPORTS
HEALTH_INSURANCE_ POLICY
PROPERTY_INSURANCE_POLICY
VEHICLE_INSURACE_POLICY
POLICY_TYPE_CODE
NAME
POLICY_TYPE_CODE
NAME
NAME
POLICY_TYPE_CODE
PROPERTY_CALIM_POLICES
COPY_OF_FIR
CLAIM ID
DATE_OF_CLAIM
INV_ARTICLE
FIRE_BRIGADE_REPORT
1
REQUIRED
1
1
REQUIRED
1
1
REQUIRED
1
HEALTH_INFO
ID
DISEASE
BLOOD_GROUP
ID
PROPERTY_INFO
TITLE_DEED
VALUE
AGE
COST_CONSTRUCTION
LOCATION
AREA
NUMBER_PLATE
Vehicle_Info
ID
MODEL
DL_NO
RC_NO
REG_YEAR

Relation Schema For Insurance Agency Management System



Functional Dependency and Normalization

1. Client (client_id, name, email, dob, city, state, pincode, bank_acc_no)
   - client_id ⟶ name
   - client_id ⟶ email
   - client_id ⟶ dob
   - client_id ⟶ state
   - client_id ⟶ city
   - client_id ⟶ pincode
   - client_id ⟶ bank_acc_no
   - pincode ⟶ state
   - pincode ⟶ city

   - **Key:-** {client_id}
   - **Prime Attribute:-** client_id, pincode
   - **BCNF** = No
     - Pincode ⟶ State and Pincode ⟶ City these two FDs violate BCNF requirement.
   - **3NF** = No
     - Same set of FDs violates 3NF requirement.
   - **2NF** = Yes

   - client_id ⟶ name
   - client_id ⟶ email
   - client_id ⟶ dob
   - client_id ⟶ pincode
   - client_id ⟶ bank_acc_no
   - pincode ⟶ state
   - pincode ⟶ city

Normalization to BCNF

- Consider relation Client (client_id, name, email, dob, city, state, pincode, bank_acc_no).If we break this relation into R1(pincode, state, city) and R2(client_id, name, email, dob, pincode, bank_acc_no).
- Based on BCNF decomposition algorithm we decompose client into R1 and R2.

- $Pincode^{+}$ = Pincode, State, City

- FD1
  - client_id ⟶ name
  - client_id ⟶ email
  - client_id ⟶ dob
  - client_id ⟶ pincode
  - client_id ⟶ bank_acc_no
- FD2
  - pincode ⟶ State
  - pincode ⟶ City

  - R1(pincode, state, city)
  - R2(client_id, name, email, dob, pincode, bank_acc_no)

2. Client's Contact (mobile_no, client_id)
   - client_id ⟶ mobile_no

   - **Key:-** { client_id }
   - **Prime Attribute:-** client_id
   - **BCNF** = Yes

3. Agent(agent_ id, name, email)
   - agent_id ⟶ name
   - agent_id ⟶ email

   - **Key:-** { agent_id }
   - **Prime Attribute:-** agent_id
   - **BCNF** = Yes

4. Agent's contact (mobile_no, agent_id)
   - agent_id ⟶ mobile_no

   - **Key:-** { agent_id }
   - **Prime Attribute:-** agent_id
   - **BCNF** = Yes

5. Policies (code, name)
   - code ⟶ name

   - **Key:-** {code}
   - **Prime Attribute:-** code
   - **BCNF** = Yes

6. Health Info (hinfo_id, disease, blood_group, policy_type_code, client_id)
   - hinfo_id ⟶ disease
   - hinfo_id ⟶ blood_group
   - hinfo_id ⟶ policy_type_code
   - hinfo_id ⟶ client_id

   - **Key:-** { hinfo_id }
   - **Prime Attribute:-** hinfo_id
   - **BCNF** = Yes

7. Property Info (pinfo_id, property_value, construction_cost, location, area, property_age, title_deed, policy_type_code, client_id)
   - pinfo_id ⟶ property_value
   - pinfo_id ⟶ construction_cost
   - pinfo_id ⟶ location
   - pinfo_id ⟶ area
   - pinfo_id ⟶ property_age
   - pinfo_id ⟶ title_deed
   - pinfo_id ⟶ policy_type_code
   - pinfo_id ⟶ client_id

   - **Key:-** { pinfo_id }
   - **Prime Attribute:-** pinfo_id
   - **BCNF** = Yes

8. Vehicle info (vinfo_id, vehicle_number, model_name, licence_no, rc_no, reg_year, policy_type_code, client_id)
   - vinfo_id ⟶ vehicle_number

- vinfo_id ⟶ model_name
- vinfo_id ⟶ licence_no
- vinfo_id ⟶ rc_no
- vinfo_id ⟶ reg_year
- vinfo_id ⟶ policy_type_code
- vinfo_id ⟶ client_id
- rc_no ⟶ vehicle_number
- licence_no ⟶ client_id
- **Key:-** { vinfo_id }
- **Prime Attribute:-** vinfo_id, rc_no, licence_no, vehicle_number
- **BCNF** = No
  - rc_no ⟶ vehicle_number and licence_no ⟶ client_id these two FDs violate BCNF requirement.
- **3NF** = No
  - Same set of FDs violate 3NF requirement.
- **2NF** = Yes

- vinfo_id ⟶ model_name
- vinfo_id ⟶ licence_no
- vinfo_id ⟶ rc_no
- vinfo_id ⟶ reg_year
- vinfo_id ⟶ policy_type_code
- rc_no ⟶ vehicle_number
- licence_no ⟶ client_id

Normalization to BCNF

- Consider relation Vehicle Info (vinfo_id, vehicle_number, model_name, licence_no, rc_no, reg_year, policy_type_code, client_id). If we break this relation into R1(vinfo_id, model_name, licence_no, rc_no, reg_year, policy_type_code) and R2(vinfo_id, rc_no, vehicle_number) and R3(vinfo_id, licence_no, client_id).
- Based on BCNF decomposition algorithm we decompose Vehicle Info into R1 and R2 and R3.

- rc_no^+ = rc_no, vehicle_number, vinfo_id
- $licence\_no^+$ = licennce_no ,client_id, vinfo_id

- FD1
  - rc_no ⟶ vehicle_number
  - rc_no ⟶ vinfo_id

- FD2
  - licence_no ⟶ client_id
  - licence_no ⟶ vinfo_id

- FD3
  - vinfo_id ⟶ model_name
  - vinfo_id ⟶ licence_no
  - vinfo_id ⟶ rc_no
  - vinfo_id ⟶ reg_year
  - vinfo_id ⟶ policy_type_code

9. Health Claim Policies (hclaim_id, medical_reports, medical_expences, date_of_claim, client_id, policy_type_code, hinfo_id)

- hclaim_id ⟶ medical_reports
- hclaim_id ⟶ medical_expences
- hclaim_id ⟶ date_of_claim
- hclaim_id ⟶ client_id
- hclaim_id ⟶ policy_type_code
- hclaim_id ⟶ hinfo_id
- medical_reports ⟶ client_id
- medical_reports ⟶ hinfo_id

- **Key:-** { hclaim_id }
- **BCNF:-** No
  - medical_reports ⟶ client_id, medical_reports ⟶ hinfo_id
    two FDs violate BCNF requirement.
- **Prime Attribute:-** hclaim_id, medical_reports
- **3NF:-** No
  - Same set of FDs violate 3NF requirement.
- **2NF:-** Yes

- hclaim_id ⟶ medical_reports
- hclaim_id ⟶ medical_expences

- hclaim_id ⟶ date_of_claim
- hclaim_id ⟶ policy_type_code
- medical_reports ⟶ client_id
- medical_reports ⟶ hinfo_id

Normalization to BCNF

- Consider relation Health Claim policies (hclaim_id, medical_reports, medical_expences, date_of_claim, client_id, policy_type_code, hinfo_id). If we break this relation into R1(hclaim_id, medical_reports, medical_expences, date_of_claim, policy_type_code) and R2(medical_reports, hinfo_id, hclaim_id).
- Based on BCNF decomposition algorithm we decompose Vehicle Info into R1 and R2.

- hclaim_id$^+$ = hclaim_id, medical_reports, medical_expences, date_of_claim, policy_type_code
- medical_reports$^+$ = medical_reports, hinfo_id, hclaim_id

- FD1
  - hclaim_id ⟶ medical_reports
  - hclaim_id ⟶ medical_expences
  - hclaim_id ⟶ date_of_claim
  - hclaim_id ⟶ policy_type_code

- FD2
  - medical_reports ⟶ client_id
  - medical_reports ⟶ hinfo_id

10. Property Claim Policies (pclaim_id, fire_brigade_report, invoice_of_article, date_of_claim, fir, pinfo_id, client_id, policy_type_code)
    - pclaim_id ⟶ fire_brigade_report
    - pclaim_id ⟶ invoice_of_article

- pclaim_id ⟶ date_of_claim
- pclaim_id ⟶ fir
- pclaim_id ⟶ pinfo_id
- pclaim_id ⟶ client_id
- pclaim_id ⟶ policy_type_code

- **Key:-** { pclaim_id }
- **Prime Attribute:-** pclaim_id
- **BCNF:-** Yes

11. Vehicle Claim Policies (vclaim_id, repaire_expense, date_of_claim, fir, vinfo_id, client_id, policy_type_code)
    - vclaim_id ⟶ repaire_expense
    - vclaim_id ⟶ date_of_claim
    - vclaim_id ⟶ fir
    - vclaim_id ⟶ vinfo_id
    - vclaim_id ⟶ client_id
    - vclaim_id ⟶ policy_type_code

    - **Key:-** { vclaim_id }
    - **Prime Attribute:-** vclaim_id
    - **BCNF:-** Yes

12. Sub Policies (sub_policy_id, sub_policies_name, description, premium, policy_period, claim_settlement_ratio, policy_type_code)
    - sub_policy_id ⟶ description
    - sub_policy_id ⟶ sub_policies_name
    - sub_policy_id ⟶ premium
    - sub_policy_id ⟶ policy_period
    - sub_policy_id ⟶ claim_settlement_ratio
    - sub_policy_id ⟶ policy_type_code

- **Key:-** { sub_policy_id }
- **Prime Attribute:-** sub_policy_id
- **BCNF:-** Yes

13. Client's Policies (client_policy_id, reg_date, exp_date, agent_id, sub_policies_id, client_id)
    - client_policy_id ⟶ reg_date
    - client_policy_id ⟶ exp_date
    - client_policy_id ⟶ agent_id
    - client_policy_id ⟶ sub_policy_id
    - client_policy_id ⟶ client_id

    - **Key:-** { client_policy_id }
    - **Prime Attribute:-** client_policy_id
    - **BCNF:-** Yes

14. Policy Payment Details (receipt_id, date_of_payment, client_policy_id, client_id)
    - receipt_id ⟶ date_of_payment
    - receipt_id ⟶ client_policy_id
    - receipt_id ⟶ client_id
    - client_policy_id ⟶ client_id

    - **Key:-** { receipt_id }
    - **Prime Attribute:-** receipt_id, client_policy_id
    - **BCNF:-** No
        - client_policy_id ⟶ client_id this FD violates BCNF requirement.
    - **3NF :-** No
        - Same FD violates 3NF requirement.
    - **2NF:-** Yes

- receipt_id ⟶ date_of_payment
- receipt_id ⟶ client_policy_id
- client_policy_id ⟶ client_id

Normalization to BCNF

- Consider relation Policy payment details (receipt_id, date_of_payment, client_policy_id, client_id). If we break this relation into R1(receipt_id, date_of_payment, client_policy_id) and R2(client_policy_id, client_id).
- Based on BCNF decomposition algorithm we decompose Vehicle Info into R1 and R2.

- receipt_id+ = receipt_id, date_of_payment, client_policy_id
- client_policy_id+ = client_policy_id, client_id

- FD1
  - receipt_id ⟶ date_of_payment
  - receipt_id ⟶ client_policy_id
- FD2
  - client_policy_id ⟶ client_id

Minimal FD Set

1. Client (client_id, name, email, dob, city, state, pincode, bank_acc_no)

Step-1: Convert RHS attribute into singleton attribute.

This FDs Is already in singleton set

- client_id ⟶ name
- client_id ⟶ email
- client_id ⟶ dob
- client_id ⟶ state
- client_id ⟶ city
- client_id ⟶ pincode
- client_id ⟶ bank_acc_no
- pincode ⟶ state
- pincode ⟶ city

Step-2: Find Closure

client_id$^+$:{client_id, name, email, dob, city, state, pincode, bank_acc_no }

- client_id ⟶ name
- client_id ⟶ email
- client_id ⟶ dob
- client_id ⟶ state
- client_id ⟶ city
- client_id ⟶ pincode
- client_id ⟶ bank_acc_no
- pincode ⟶ state
- pincode ⟶ city

Step-3: Remove Redundant FDs.

- client_id ⟶ name
- client_id ⟶ email
- client_id ⟶ dob
- client_id ⟶ pincode
- client_id ⟶ bank_acc_no
- pincode ⟶ state
- pincode ⟶ city

- client_id ⟶ name,email, dob, city, state, pincode, bank_acc_no
- pincode ⟶ state,city

2. Client's Contact (mobile_no, client_id)
   - client_id ⟶ mobile_no

   This Fds is in Minimal-FDs.

3. Agent(agent_ id, name, email)
   - agent_id ⟶ name
   - agent_id ⟶ email

This RHS attribute is already in singleton Attribute

Step-2: Find Closure

Agent_id$^+$={agent_id,name,email}

- agent_id ⟶ name
- agent_id ⟶ email

Step-3: Remove Redundant FDs

- agent_id ⟶ name,email

4. Agent's contact (mobile_no, agent_id)
   - agent_id ⟶ mobile_no

   This Fds is in Minimal-FDs.

5. Policies (code, name)
   - code ⟶ name

   This Fds is in Minimal-FDs.

6. Health Info (hinfo_id, disease, blood_group, policy_type_code, client_id)
   - hinfo_id ⟶ disease
   - hinfo_id ⟶ blood_group
   - hinfo_id ⟶ policy_type_code
   - hinfo_id ⟶ client_id

This RHS attribute is already in singleton Attribute.

Step-2: Find Closure

Health Info$^+$={hinfo_id, disease, blood_group, policy_type_code, client_id}

- hinfo_id ⟶ disease
- hinfo_id ⟶ blood_group
- hinfo_id ⟶ policy_type_code
- hinfo_id ⟶ client_id

Step-3: Remove Redundant FDs

- hinfo_id ⟶ disease, blood_group, policy_type_code, client_id

7. Property Info (pinfo_id, property_value, construction_cost, location, area, property_age, title_deed, policy_type_code, client_id)
   - pinfo_id ⟶ property_value
   - pinfo_id ⟶ construction_cost
   - pinfo_id ⟶ location
   - pinfo_id ⟶ area
   - pinfo_id ⟶ property_age
   - pinfo_id ⟶ title_deed
   - pinfo_id ⟶ policy_type_code
   - pinfo_id ⟶ client_id

   This RHS attribute is already in singleton Attribute.

Step-2: Find Closure

pinfo_id$^+$={property_value, construction_cost, location, area, property_age, title_deed, policy_type_code, client_id}

- pinfo_id ⟶ property_value
- pinfo_id ⟶ construction_cost
- pinfo_id ⟶ location
- pinfo_id ⟶ area
- pinfo_id ⟶ property_age
- pinfo_id ⟶ title_deed
- pinfo_id ⟶ policy_type_code
- pinfo_id ⟶ client_id

Step-3: Remove Redundant FDs

- pinfo_id ⟶ property_value, construction_cost, location, area, property_age, title_deed, policy_type_code, client_id

8. Vehicle info (vinfo_id, vehicle_number, model_name, licence_no, rc_no, reg_year, policy_type_code, client_id)
   - vinfo_id ⟶ vehicle_number
   - vinfo_id ⟶ model_name
   - vinfo_id ⟶ licence_no
   - vinfo_id ⟶ rc_no
   - vinfo_id ⟶ reg_year
   - vinfo_id ⟶ policy_type_code
   - vinfo_id ⟶ client_id
   - rc_no ⟶ vehicle_number
   - licence_no ⟶ client_id

Step-1: Convert RHS attribute into singleton attribute.

- vinfo_id ⟶ vehicle_number
- vinfo_id ⟶ model_name
- vinfo_id ⟶ licence_no
- vinfo_id ⟶ rc_no
- vinfo_id ⟶ reg_year
- vinfo_id ⟶ policy_type_code
- vinfo_id ⟶ client_id
- rc_no ⟶ vehicle_number
- licence_no ⟶ client_id

Step-2: Find Closure

$vinfo\_id^+$={vehicle_number, model_name, licence_no, rc_no, reg_year, policy_type_code, client_id}

- vinfo_id ⟶ vehicle_number
- vinfo_id ⟶ model_name
- vinfo_id ⟶ licence_no
- vinfo_id ⟶ rc_no
- vinfo_id ⟶ reg_year
- vinfo_id ⟶ policy_type_code
- vinfo_id ⟶ client_id

- rc_no ⟶ vehicle_number
- licence_no ⟶ client_id

Step-3: Remove Redundant FDs

- vinfo_id ⟶ model_name
- vinfo_id ⟶ licence_no
- vinfo_id ⟶ rc_no
- vinfo_id ⟶ reg_year
- vinfo_id ⟶ policy_type_code
- rc_no ⟶ vehicle_number
- licence_no ⟶ client_id

- vinfo_id ⟶ vehicle_number, model_name, licence_no, rc_no, reg_year, policy_type_code, client_id
- rc_no ⟶ vehicle_number
- licence_no ⟶ client_id

9. Health Claim Policies (hclaim_id, medical_reports, medical_expences, date_of_claim, client_id, policy_type_code, hinfo_id)
   - hclaim_id ⟶ medical_reports
   - hclaim_id ⟶ medical_expences
   - hclaim_id ⟶ date_of_claim
   - hclaim_id ⟶ client_id
   - hclaim_id ⟶ policy_type_code
   - hclaim_id ⟶ hinfo_id
   - medical_reports ⟶ client_id
   - medical_reports ⟶ hinfo_id

Step-1: Convert RHS attribute into singleton attribute.

- hclaim_id ⟶ medical_reports
- hclaim_id ⟶ medical_expences
- hclaim_id ⟶ date_of_claim
- hclaim_id ⟶ client_id
- hclaim_id ⟶ policy_type_code
- hclaim_id ⟶ hinfo_id
- medical_reports ⟶ client_id
- medical_reports ⟶ hinfo_id

Step-2: Find Closure

hclaim_id$^+$={hclaim_id,medical_reports,medical_expences,date_of_claim, client_id, policy_type_code, hinfo_id}

- hclaim_id ⟶ medical_reports
- hclaim_id ⟶ medical_expences
- hclaim_id ⟶ date_of_claim
- hclaim_id ⟶ client_id
- hclaim_id ⟶ policy_type_code
- hclaim_id ⟶ hinfo_id
- medical_reports ⟶ client_id
- medical_reports ⟶ hinfo_id

Step-3: Remove Redundant FDs

- hclaim_id ⟶ medical_reports
- hclaim_id ⟶ medical_expences
- hclaim_id ⟶ date_of_claim
- hclaim_id ⟶ policy_type_code
- medical_reports ⟶ client_id
- medical_reports ⟶ hinfo_id

- hclaim_id ⟶ medical_reports, medical_reports, medical_expences, date_of_claim, client_id, policy_type_code, hinfo_id
- medical_reports ⟶ client_id,hinfo_id

10. Property Claim Policies (pclaim_id, fire_brigade_report, invoice_of_article, date_of_claim, fir, pinfo_id, client_id, policy_type_code)
    - pclaim_id ⟶ fire_brigade_report
    - pclaim_id ⟶ invoice_of_article
    - pclaim_id ⟶ date_of_claim
    - pclaim_id ⟶ fir
    - pclaim_id ⟶ pinfo_id
    - pclaim_id ⟶ client_id
    - pclaim_id ⟶ policy_type_code

Step-1: Convert RHS attribute into singleton attribute

- pclaim_id → fire_brigade_report
- pclaim_id → invoice_of_article
- pclaim_id → date_of_claim
- pclaim_id → fir
- pclaim_id → pinfo_id
- pclaim_id → client_id
- pclaim_id → policy_type_code

Step-2: Find Closure

pclaim_id$^+$={ pclaim_id,fire_brigade_report, invoice_of_article, date_of_claim, fir, pinfo_id, client_id, policy_type_code}

- pclaim_id → fire_brigade_report
- pclaim_id → invoice_of_article
- pclaim_id → date_of_claim
- pclaim_id → fir
- pclaim_id → pinfo_id
- pclaim_id → client_id
- pclaim_id → policy_type_code

Step-3: Remove Redundant FDs

- pclaim_id → fire_brigade_report,invoice_of_article, date_of_claim, fir, pinfo_id, client_id, policy_type_code

11. Vehicle Claim Policies (vclaim_id, repaire_expense, date_of_claim, fir, vinfo_id, client_id, policy_type_code)
    - vclaim_id → repaire_expense
    - vclaim_id → date_of_claim
    - vclaim_id → fir
    - vclaim_id → vinfo_id
    - vclaim_id → client_id
    - vclaim_id → policy_type_code

Step-1: Convert RHS attribute into singleton attribute

- vclaim_id → repaire_expense
- vclaim_id → date_of_claim

- vclaim_id ⟶ fir
- vclaim_id ⟶ vinfo_id
- vclaim_id ⟶ client_id
- vclaim_id ⟶ policy_type_code

Step-2: Find Closure

vclaim_id$^+$={vclaim_id, repaire_expense, date_of_claim, fir, vinfo_id, client_id, policy_type_code}

- vclaim_id ⟶ repaire_expense
- vclaim_id ⟶ date_of_claim
- vclaim_id ⟶ fir
- vclaim_id ⟶ vinfo_id
- vclaim_id ⟶ client_id
- vclaim_id ⟶ policy_type_code

Step-3: Remove Redundant FDs

- vclaim_id ⟶ repaire_expense
- vclaim_id ⟶ date_of_claim
- vclaim_id ⟶ fir
- vclaim_id ⟶ vinfo_id
- vclaim_id ⟶ client_id
- vclaim_id ⟶ policy_type_code

- vclaim_id ⟶ repaire_expense, date_of_claim, fir, vinfo_id, client_id, policy_type_code

12. Sub Policies (sub_policy_id, sub_policies_name, description, premium, policy_period, claim_settlement_ratio, policy_type_code)
    - sub_policy_id ⟶ description
    - sub_policy_id ⟶ sub_policies_name
    - sub_policy_id ⟶ premium
    - sub_policy_id ⟶ policy_period
    - sub_policy_id ⟶ claim_settlement_ratio
    - sub_policy_id ⟶ policy_type_code

Step-1: Convert RHS attribute into singleton attribute

- sub_policy_id ⟶ description
- sub_policy_id ⟶ sub_policies_name
- sub_policy_id ⟶ premium

- sub_policy_id ⟶ policy_period
- sub_policy_id ⟶ claim_settlement_ratio
- sub_policy_id ⟶ policy_type_code

Step-2: Find Closure

sub_policy_id$^+$={sub_policy_id,description,premium,policy_period, claim_settlement_ratio, policy_type_code, sub_policies_name }

- sub_policy_id ⟶ description
- sub_policy_id ⟶ sub_policies_name
- sub_policy_id ⟶ premium
- sub_policy_id ⟶ policy_period
- sub_policy_id ⟶ claim_settlement_ratio
- sub_policy_id ⟶ policy_type_code

Step-3: Remove Redundant FDs

- sub_policy_id ⟶ description
- sub_policy_id ⟶ sub_policies_name
- sub_policy_id ⟶ premium
- sub_policy_id ⟶ policy_period
- sub_policy_id ⟶ claim_settlement_ratio
- sub_policy_id ⟶ policy_type_code

- sub_policy_id, sub_policies_name,description,premium,policy_period, claim_settlement_ratio, policy_type_code

13. Client's Policies (client_policy_id, reg_date, exp_date, agent_id, sub_policies_id, client_id)
    - client_policy_id ⟶ reg_date
    - client_policy_id ⟶ exp_date
    - client_policy_id ⟶ agent_id
    - client_policy_id ⟶ sub_policy_id
    - client_policy_id ⟶ client_id

Step-1: Convert RHS attribute into singleton attribute

- client_policy_id ⟶ reg_date
- client_policy_id ⟶ exp_date
- client_policy_id ⟶ agent_id
- client_policy_id ⟶ sub_policy_id

- client_policy_id ⟶ client_id

Step-2: Find Closure

client_policy_id$^+$={client_policy_id,reg_date,exp_date,agent_id, sub_policies_id, client_id}

- client_policy_id ⟶ reg_date
- client_policy_id ⟶ exp_date
- client_policy_id ⟶ agent_id
- client_policy_id ⟶ sub_policy_id
- client_policy_id ⟶ client_id

Step-3: Remove Redundant FDs

- client_policy_id ⟶ reg_date
- client_policy_id ⟶ exp_date
- client_policy_id ⟶ agent_id
- client_policy_id ⟶ sub_policy_id
- client_policy_id ⟶ client_id

- client_policy_id ⟶ reg_date, exp_date, agent_id, sub_policies_id, client_id

14. Policy Payment Details (receipt_id, date_of_payment, client_policy_id, client_id)
    - receipt_id ⟶ date_of_payment
    - receipt_id ⟶ client_policy_id
    - receipt_id ⟶ client_id
    - client_policy_id ⟶ client_id

Step-1: Convert RHS attribute into singleton attribute

- receipt_id ⟶ date_of_payment
- receipt_id ⟶ client_policy_id
- receipt_id ⟶ client_id
- client_policy_id ⟶ client_id

Step-2: Find Closure

receipt_id$^+$={ receipt_id, date_of_payment, client_policy_id, client_id}

- receipt_id ⟶ date_of_payment
- receipt_id ⟶ client_policy_id
- receipt_id ⟶ client_id
- client_policy_id ⟶ client_id

Step-3: Remove Redundant FDs

- receipt_id ⟶ date_of_payment
- receipt_id ⟶ client_policy_id
- client_policy_id ⟶ client_id

- receipt_id ⟶ date_of_payment, client_policy_id, client_id
- client_policy_id ⟶ client_id

DDL Script For Insurance Agency Management System

```
CREATE TABLE CLIENT (
   Client_id INTEGER,
   Name VARCHAR(30),
   Email VARCHAR(30),
   DOB DATE,
   City VARCHAR(20),
   State VARCHAR(20),
   Pincode INTEGER,
   Bank_acc_no DECIMAL(16),
   PRIMARY KEY(Client_id)
   );

CREATE TABLE CLIENT_CONTACT (
   Client_id_ref INTEGER,
   Mobile_no BIGINT,
   FOREIGN KEY(Client_id_ref) REFERENCES CLIENT(Client_id)
               ON DELETE SET NULL ON UPDATE CASCADE,
   PRIMARY KEY(Client_id_ref,Mobile_no)
   );


CREATE TABLE Agents (
   Agent_id SMALLINT PRIMARY KEY,
   Name VARCHAR(30),
   Email VARCHAR(30)
   );

CREATE TABLE AGENT_CONTACT(
  Mobile_no BIGINT,
  Agent_id_ref SMALLINT,
  FOREIGN KEY (Agent_id_ref) REFERENCES Agents(Agent_id)
        ON DELETE CASCADE ON UPDATE CASCADE,
  PRIMARY KEY (Mobile_no,Agent_id_ref)
  );


CREATE TABLE POLICIES (
   Code INTEGER,
   Name VARCHAR(20),
   PRIMARY KEY (Code)
   );
```

```
CREATE TABLE SUB_POLICIES (
   Sub_policies_id SMALLINT,
   Sub_policies_name VARCHAR(30),
   Policy_type_code_ref SMALLINT,
   Description VARCHAR(50),
   Premium FLOAT,
   Policy_period INTEGER,
   Claim_settlement_ratio FLOAT,
   PRIMARY KEY (Sub_policies_id)
   FOREIGN KEY (Policy_type_code_ref) REFERENCES POLICIES (Code)
       ON DELETE SET NULL ON UPDATE CASCADE
 );

CREATE TABLE Client_policies (
   Client_policy_id SMALLINT PRIMARY KEY,
   Reg_date DATE,
   Exp_date DATE,
   Agent_id_ref SMALLINT,
   Sub_policies_id_ref SMALLINT,
   Client_id_ref SMALLINT,
   FOREIGN KEY (Agent_id_ref) REFERENCES Agents (Agent_id)
       ON DELETE CASCADE ON UPDATE CASCADE,
   FOREIGN KEY (Client_id_ref) REFERENCES CLIENT (Client_id)
       ON DELETE CASCADE ON UPDATE CASCADE,
   FOREIGN KEY (Sub_policies_id_ref) REFERENCES SUB_POLICIES
(Sub_policies_id)
       ON DELETE CASCADE ON UPDATE CASCADE
   );


CREATE TABLE Policy_payment_details (
   Reciept_id NUMERIC(10,0) PRIMARY KEY,
   Date_of_payment DATE,
   Client_policy_id_ref SMALLINT,
   Client_id_ref SMALLINT,
   FOREIGN KEY (Client_id_ref) REFERENCES CLIENT (Client_id)
       ON DELETE CASCADE ON UPDATE CASCADE,
   FOREIGN KEY (Client_policy_id_ref) REFERENCES SUB_POLICIES
(Sub_policies_id)
       ON DELETE SET NULL ON UPDATE CASCADE
```

```
);
CREATE TABLE SALES(
   Client_id_ref SMALLINT,
   Agent_id_ref SMALLINT,
   Policies_code_ref INTEGER,
   FOREIGN KEY (Client_id_ref) REFERENCES CLIENT (Client_id)
        ON DELETE CASCADE ON UPDATE CASCADE,
   FOREIGN KEY (Agent_id_ref) REFERENCES AGENTS (Agent_id)
        ON DELETE CASCADE ON UPDATE CASCADE,
   FOREIGN KEY (Policies_code_ref) REFERENCES POLICIES (Code)
        ON DELETE SET NULL ON UPDATE CASCADE
   );



CREATE TABLE Vehicle_Info (
   Vinfo_ID SMALLINT PRIMARY KEY,
   Vehicle_no VARCHAR(10),
   Model_name VARCHAR(50),
   Licence_no VARCHAR(15),
   RC_no VARCHAR(10),
   Reg_year INT,
   Sub_policies_id_ref SMALLINT,
   Client_id_ref SMALLINT,
   FOREIGN KEY (Sub_policies_id_ref) REFERENCES SUB_POLICIES
(Sub_policies_id),
        ON DELETE CASCADE ON UPDATE CASCADE,
   FOREIGN KEY (Client_id_ref) REFERENCES CLIENT(Client_id)
        ON DELETE CASCADE ON UPDATE CASCADE,
);



CREATE TABLE Vehicle_Claim_Policies (
   Vclaim_id SMALLINT PRIMARY KEY,
   Repaire_expense DECIMAL(9,2),
   Date_of_claim DATE,
   FIR VARCHAR(10),
   VInfo_ID_ref SMALLINT,
   Client_id_ref SMALLINT,
   Sub_policies_id_ref SMALLINT,
   FOREIGN KEY(VInfo_ID_ref) REFERENCES Vehicle_Info(Vinfo_ID)
```

```
        ON DELETE CASCADE ON UPDATE CASCADE,
    FOREIGN KEY (Client_id_ref) REFERENCES CLIENT(Client_id)
        ON DELETE CASCADE ON UPDATE CASCADE,
FOREIGN KEY (Sub_policies_id_ref) REFERENCES SUB_POLICIES
(Sub_policies_id),
        ON DELETE CASCADE ON UPDATE CASCADE
    );



CREATE TABLE Health_info (
    Hinfo_id SMALLINT PRIMARY KEY,
    Disease VARCHAR(100),
    Blood_group VARCHAR(3),
    Sub_policies_id_ref SMALLINT,
    Client_id_ref SMALLINT,
    FOREIGN KEY (Client_id_ref) REFERENCES CLIENT(Client_id)
        ON DELETE CASCADE ON UPDATE CASCADE,
    FOREIGN KEY (Sub_policies_id_ref) REFERENCES SUB_POLICIES
(Sub_policies_id)
        ON DELETE CASCADE ON UPDATE CASCADE
);



CREATE TABLE Health_claim_policy (
    Hclaim_id SMALLINT PRIMARY KEY,
    Medical_reports VARCHAR(10),
    Medical_expences DECIMAL(9,2),
    Date_of_claim DATE,
    Client_id_ref SMALLINT,
    Sub_policies_id_ref SMALLINT,
    Hinfo_id_ref SMALLINT,
    FOREIGN KEY (Client_id_ref) REFERENCES CLIENT(Client_id)
        ON DELETE CASCADE ON UPDATE CASCADE,
    FOREIGN KEY (Sub_policies_id_ref) REFERENCES SUB_POLICIES
(Sub_policies_id)
        ON DELETE CASCADE ON UPDATE CASCADE,
    FOREIGN KEY (Hinfo_id_ref) REFERENCES Health_info (Hinfo_id)
        ON DELETE CASCADE ON UPDATE CASCADE
    );
```

```
CREATE TABLE Property_info (
   Pinfo_id SMALLINT PRIMARY KEY,
   Property_value DECIMAL(11,2),
   Construction_cost DECIMAL(11,2),
   Area DECIMAL(5,3),
   Location VARCHAR(30),
   Property_age INT,
   Title_deed VARCHAR(10),
   Sub_policies_id_ref SMALLINT,
   Client_id_ref SMALLINT,
   FOREIGN KEY (Client_id_ref) REFERENCES CLIENT(Client_id)
        ON DELETE CASCADE ON UPDATE CASCADE,
   FOREIGN KEY (Sub_policies_id_ref) REFERENCES SUB_POLICIES
(Sub_policies_id)
        ON DELETE CASCADE ON UPDATE CASCADE
);
```

```
CREATE TABLE Property_claim_policies (
   Pclaim_id SMALLINT PRIMARY KEY,
   Fire_brigade_report VARCHAR(10),
   Invoice_of_artical DECIMAL(9,2),
   Date_of_claim DATE,
   FIR VARCHAR(10),
   Pro_info_id_ref SMALLINT,
   Client_id_ref SMALLINT,
   Sub_policies_id_ref SMALLINT,
   FOREIGN KEY (Pro_info_id_ref) REFERENCES Property_info (Pinfo_id)
        ON DELETE CASCADE ON UPDATE CASCADE,
   FOREIGN KEY (Client_id_ref) REFERENCES CLIENT (Client_id)
        ON DELETE CASCADE ON UPDATE CASCADE,
   FOREIGN KEY (Sub_policies_id_ref) REFERENCES SUB_POLICIES
(Sub_policies_id)
        ON DELETE CASCADE ON UPDATE CASCADE
   );
```

Insert Queries For Insurance Agency Management System

- client Table

```
insert into client values
(1001,'Rohit','rh@gmail.com','1965-05-
05','Hyderabad','Telangana','500008','7032883964450001'),
(1002,'Sharma','sharmaji@gmail.com','1966-06-
06','Jaipur','Rajasthan','302001','7032883964450002'),
(1003,'Virat','vk@gmail.com','1970-10-
10','Delhi','Delhi','110001','7032883964450003'),
(1004,'Surya','sky@gmail.com','1980-12-
10','Banglore','Karnataka','530068','7032883964450004'),
(1005,'Panchal','panchal@gmail.com','1971-10-
10','Surat','Gujarat','335009','7032883964450005'),
(1006,'Mahendra','msd@gmail.com','1972-01-
01','Chennai','Tamilnadu','600001','7032883964450006'),
(1007,'Rahul','rh@gmail.com','1973-02-
03','Pune','Maharashtra','411002','7032883964450007'),
(1008,'Umar','um@gmail.com','1974-05-
10','Lucknow','UttarPradesh','226001','7032883964450008'),
(1009,'Faruk','fk@gmail.com','1970-04-
04','Bhopal','Madhyapradesh','462001','7032883964450009'),
(1010,'Mithali','mt@gmail.com','1980-02-
25','Vishakhapatnam','Andhrapradesh','530001','7032883964450010'),
(1011,'Feni','fen@gmail.com','1990-12-
10','Patna','Bihar','800001','7032883964450011'),
(1012,'Smriti','sm@gmail.com','1995-10-
04','Ghaziabad','Uttarpradesh','201003','7032883964450012'),
(1013,'Sania','ss@gmail.com','1972-09-
19','Ludhiana','Punjab','141003','7032883964450013'),
(1014,'Anushka','as@gmail.com','1980-12-
10','Varanasi','Uttarpradesh','212011','7032883964450014'),
(1015,'Bhuvneshwar','bk@gmail.com','1945-07-
17','Salem','Tamilnadu','636002','7032883964450015');
```

- client_contact Table

```
insert into client_contact values
(1001,7555476832),
(1002,8555242652),
(1003,9555514393),
```

(1004,9955571582),
(1005,9555523566),
(1006,8555325677),
(1007,9255589513),
(1008,7555184489),
(1009,7555268345),
(1010,9855534051),
(1011,7555733357),
(1012,9455580389),
(1013,8555926426),
(1014,7555947444),
(1015,9855594515);

- agents Table

insert into agents values
(5101,'Salim','salim@gmail.com'),
(5102,'Manoj','manoj@gmail.com'),
(5103,'Vatsal','vt@gmail.com'),
(5104,'Anjali','atm@gmail.com'),
(5105,'Dev','d12@gmail.com');

- agents_contact Table

insert into agent_contact values
(7654123842,5101),
(8457561232,5102),
(9898989898,5103),
(9636469669,5104),
(7887375787,5105);

- policies table

insert into policies values
(1001,'Health Insurance'),
(1002,'Property Insurance'),
(1003,'Vehicle Insurance');

- Sub_Policies Table

```
INSERT INTO sub_policies VALUES
(9001,'Individual',1001,'5 Lakh Cover',34255.00,1,95.5),
(9002,'Corona Kavach',1001,'5.5 Lakh Cover',8409.54,3,89.1),
(9003,'Home',1002,'5 Cr Cover',36285.00,3,87.0),
(9004,'Commercial',1002,'6 Cr Cover',39356,3,86.5),
(9005,'Car',1003,'4,64,835 IDV',15104.00,1,80.0),
(9006,'Bike',1003,'51,824 IDV',240.35,1,90.0);
```

- Health_info Table

```
insert into health_info values
(301,'Diabetes','AB+',9001,1003),
(302,'High Blood Pressure','B+',9001,1001),
(303, 'Cholesterol','A+',9002,1007),
(304,'Uric Acid','AB-',9002,1010),
(305,'Asthma','B-',9002,1002),
(306,'Pneumonia','B+',9002,1004),
(307,'Strep throat','AB+',9001,1006);
```

- Property_info Table

```
insert into property_info values
(2001,70000000,20000000,'18,000 sq.ft','Punjab',5,'Yes',9004,1013),
(2002,50000000,10000000,'15,000 sq.ft','Uttarpradesh',4,'Yes',9004,1012),
(2003,30000000,9500000,'12,000 sq.ft','Madhyapradesh',3,'Yes',9003,1009),
(2004,20000000,8100000,'9,000 sq.ft','Tamilnadu',2,'Yes',9003,1015),
(2005,6000000,7000000,'5,000 sq.ft','Bihar',3,'Yes',9003,1011),
(2006,20000000,8000000,'6000 sq.ft','Delhi',4,'Yes',9003,1003),
(2007,40000000,12000000,'9000 sq.ft','Uttarpradesh',5,'Yes',9004,1008);
```

- Vehicle_info Table

```
insert into vehicle_info values
(4001,'TN01AB1235','Hellcat
X132','TN0120151203654','TN01ZQ1256',2019,9006,1006),
```

(4002,'KA02FG5689','Kawasaki
Z900','KA0420142032563','KA02RT6321',2017,9006,1004),
(4003,'UP03JK5202','Honda
Amaez','UP0520162023698','UP03SD4257',2016,9005,1014),
(4004,'GJ07LM1456','Hyundai
Creta','GJ0720194563210','GJ07IO6938',2015,9005,1005),
(4005,'UP06FD8765','Volkswagen Tiguan
Allspace','UP0920186325874','UP06YU2697',2020,9005,1008),
(4006,'TS02CD2346','Baleno','TS1224562314765','TS02AS2367',2018,9005,10
01);

- client policies

INSERT INTO client_policies VALUES
(01,'2019-02-12','2020-02-12',5101,9001,1001),
(02,'2021-01-15','2022-01-15',5102,9001,1003),
(03,'2019-08-21','2022-08-21',5105,9002,1002),
(04,'2020-06-10','2023-06-10',5101,9002,1007),
(05,'2021-01-01','2024-01-01',5105,9002,1010),
(06,'2015-08-20','2018-08-20',5101,9003,1009),
(07,'2018-07-06-','2021-07-06',5102,9003,1011),
(08,'2021-03-08','2024-03-08',5102,9003,1015),
(09,'2014-04-10','2017-04-10',5104,9004,1012),
(10,'2021-09-01','2024-09-01',5104,9004,1013),
(11,'2016-01-15','2017-01-15',5104,9005,1005),
(12,'2021-10-21','2022-10-21',5101,9005,1008),
(13,'2018-12-12','2019-12-12',5102,9005,1014),
(14,'2019-07-07','2020-07-07',5103,9006,1004),
(15,'2020-02-20','2021-02-20',5103,9006,1006),
(16,'2020-02-12','2021-02-12',5103,9005,1001),
(17,'2018-10-18','2021-10-18',5102,9003,1003),
(18,'2019-05-11','2020-05-11',5104,9002,1004),
(19,'2017-06-20','2020-06-20',5101,9004,1008),
(20,'2020-05-17','2021-05-17',5105,9001,1006);

- Health_claim_policies

```
INSERT INTO health_claim_policy VALUES
(11101,'Yes',150000,'2019-11-15',1001,9001,302),
(11102,'Yes',350000,'2021-09-20',1003,9001,301),
(11103,'Yes',200000,'2020-10-31',1002,9002,305),
(11104,'Yes',300000,'2021-05-18',1007,9002,303),
(11105,'Yes',180000,'2019-11-18',1004,9002,306),
(11106,'Yes',200000,'2020-10-20',1006,9001,307);
```

- Property_claim_policies

```
INSERT INTO property_claim_policies VALUES
(11201,'Yes',15000000,'2017-05-23','Yes',2003,1009,9003),
(11202,'Yes',20000000,'2020-04-15','Yes',2005,1011,9003),
(11203,'Yes',10000000,'2016-08-19','Yes',2002,1012,9004),
(11204,'Yes',28000000,'2021-11-30','Yes',2001,1013,9004),
(11205,'Yes',7000000,'2020-08-01','Yes',2006,1003,9003),
(11206,'Yes',20000000,'2019-07-17','Yes',2007,1008,9004);
```

- Vehicle_claim_policies

```
INSERT INTO vehicle_claim_policies VALUES
(11301,50000,'2016-11-30','Yes',4004,1005,9005),
(11302,150000,'2019-09-15','Yes',4003,1014,9005),
(11303,21000,'2020-02-01','Yes',4002,1004,9006),
(11304,120000,'2020-11-05','Yes',4006,1001,9005);
```

- policy_payment_details

```
INSERT INTO policy_payment_details VALUES
(101,'2019-02-12',1,1001),
(102,'2021-01-15',2,1003),
(103,'2019-08-21',3,1002),
(104,'2020-06-10',4,1007),
(105,'2021-01-01',5,1010),
(106,'2015-08-20',6,1009),
(107,'2018-07-06',7,1011),
(108,'2021-03-08',8,1015),
(109,'2014-04-10',9,1012),
(110,'2021-09-01',10,1013),
(111,'2016-01-15',11,1005),
(112,'2021-10-21',12,1008),
(113,'2018-12-12',13,1014),
(114,'2019-07-07',14,1004),
(115,'2020-02-20',15,1006),
(116,'2020-02-12',16,1001),
(117,'2018-10-18',17,1003),
(118,'2017-06-20',18,1008),
(119,'2019-05-11',19,1004),
(120,'2020-05-17',20,1006);
```

SQL Queries For Insurance Agency Management System

1.List all the clients who have taken Bike Vehicle insurance.

- SELECT c1.client_id, c1.name, cp.sub_policies_id_ref FROM client AS c1 JOIN client_policies AS cp ON c1.client_id = cp.client_id_ref JOIN sub_policies AS s1 ON s1.sub_policies_id = cp.sub_policies_id_ref WHERE s1.sub_policies_name = 'Bike';

- **OUTPUT: -**

Explain Notifications Messages Data Output

	client_id integer	name character varying (30)	sub_policies_id_ref smallint
1	1004	Surya	9006
2	1006	Mahendra	9006

2. List all clients who have taken the policy code 1001 and have validity greater than 1 year.

- SELECT c1.client_id, c1.name, p1.code, sp.policy_period FROM client AS c1 JOIN client_policies AS cp
  ON c1.client_id = cp.client_id_ref JOIN sub_policies AS sp
  ON cp.sub_policies_id_ref = sp.sub_policies_id JOIN policies AS p1
  ON sp.policy_type_code_ref = p1.code
  WHERE p1.code = 1001 AND sp.policy_period > 1;

- **OUTPUT: -**

Explain Notifications Messages Data Output

	client_id integer	name character varying (30)	code integer	policy_period integer
1	1002	Sharma	1001	3
2	1007	Rahul	1001	3
3	1010	Mithali	1001	3

3. List all clients who have taken policy from agent's ID 5103.

- SELECT c1.client_id, c1.name, a1.agent_id, a1.name, a1.email FROM client AS c1 JOIN client_policies AS cp
  ON c1.client_id = cp.client_id_ref JOIN agents AS a1
  ON cp.agent_id_ref = a1.agent_id

WHERE a1.agent_id = 5103;

- **OUTPUT: -**

Explain Notifications Messages Data Output

	client_id integer	name character varying (30)	agent_id smallint	name character varying (30)	email character varying (30)
1	1004	Surya	5103	Vatsal	vt@gmail.com
2	1006	Mahendra	5103	Vatsal	vt@gmail.com

4. List out all the client ID with phone number who has taken Health policy from agent ID 5101.

- SELECT c1.client_id, c1.name, cc.mobile_no, p1.name FROM client AS c1 JOIN client_contact AS cc
  ON c1.client_id = cc.client_id_ref JOIN client_policies AS cp
  ON cc.client_id_ref = cp.client_id_ref JOIN sub_policies AS sp
  ON cp.sub_policies_id_ref = sp.sub_policies_id JOIN policies p1
  ON sp.policy_type_code_ref = p1.code
  WHERE p1.name = 'Health Insurance';

- **OUTPUT: -**

Explain Notifications Messages Data Output

	client_id integer	name character varying (30)	mobile_no bigint	name character varying (20)
1	1001	Rohit	7555476832	Health Insurance
2	1003	Virat	9555514393	Health Insurance
3	1002	Sharma	8555242652	Health Insurance
4	1007	Rahul	9255589513	Health Insurance
5	1010	Mithali	9855534051	Health Insurance

5. List all clients (client's id, Name, Email) who have taken more than one policy.

- select client_id,name,email from client as c
  except
  (select client_id,name,email from client as c join client_policies as cp on c.client_id=cp.client_id_ref

group by client_id,name,email having count(client_id)<2) order by client_id asc;

- **OUTPUT: -**

Explain Notifications Messages Data Output

	client_id [PK] integer	name character varying (30)
1	1001	Rohit
2	1003	Virat
3	1004	Surya
4	1006	Mahendra
5	1008	Umar

6. Give all the payment details of client ID 1008.

- SELECT pd.reciept_id, pd.date_of_payment, pd.client_id_ref, cp.sub_policies_id_ref FROM policy_payment_details AS pd JOIN client_policies AS cp
  ON pd.client_policy_id_ref = cp.client_policy_id
  WHERE pd.client_id_ref = 1008;

- **OUTPUT: -**

Explain Notifications Messages Data Output

	reciept_id numeric (10)	date_of_payment date	client_id_ref smallint	sub_policies_id_ref smallint
1	112	2021-10-21	1008	9005
2	118	2017-06-20	1008	9002

7. Update claim settlement ratio to 98.9% of sub policy ID 9002.

- UPDATE sub_policies SET claim_settlement_ratio = 98.9
  WHERE sub_policies_id = 9002
  RETURNING *;

- **OUTPUT: -**

Explain Notifications Messages Data Output

	sub_policies_id [PK] smallint	sub_policies_name character varying (30)	policy_type_code_ref smallint	description character varying (100)	premium double precision	policy_period integer	claim_settlement_ratio double precision
1	9002	Corona Kavach	1001	5.5 Lakh Cover	8409.54	3	98.9

8. List all the sub-policies (id, policy name, description) of policy name=’Property Insurance’.

- SELECT * FROM sub_policies AS sp JOIN policies AS p1
  ON sp.policy_type_code_ref = p1.code
  WHERE p1.name = 'Property Insurance';

- **OUTPUT: -**

Explain Notifications Messages Data Output

	sub_policies_id smallint	sub_policies_name character varying (30)	policy_type_code_ref smallint	description character varying (100)	premium double precision	policy_period integer	claim_settlement_ratio double precision	code integer	name character varying (20)
1	9003	Home	1002	5 Cr Cover	36285	3	87	1002	Property Insurance
2	9004	Commercial	1002	6 Cr Cover	39356	3	86.5	1002	Property Insurance

9. Retrieve date of registration and date of expiry of the policy of a client who has taken Health policy from agent ID 5101.

- select reg_date,exp_date,agent_id_ref,client_id_ref from client_policies as cp join sub_policies as sp
  on cp.sub_policies_id_ref=sp.sub_policies_id join policies as p
  on sp.policy_type_code_ref=p.code
  where p.name='Health Insurance' and cp.agent_id_ref=5101;

- **OUTPUT: -**

Explain Notifications Messages Data Output

	reg_date date	exp_date date	agent_id_ref smallint	client_id_ref smallint
1	2019-02-12	2020-02-12	5101	1001
2	2020-06-10	2023-06-10	5101	1007

10. Most valuable agent who sold the most number of policies.

- SELECT a.agent_id,count(client_policy_id) AS NO_OF_SALE from agents AS a INNER JOIN client_policies AS cp
  ON a.agent_id = cp.agent_id_ref
  GROUP BY a.agent_id ORDER BY NO_OF_SALE DESC LIMIT 1;

- **OUTPUT: -**

Explain Notifications Messages Data Output

	agent_id [PK] smallint	no_of_sale bigint
1	5102	5

11. List all the policy names with information of the client named 'Rahul'.

- Select
  client_id,name,email,reg_date,exp_date,sub_policies_id_ref,agent_id_ref
  from client as c join client_policies as cp
  on c.client_id=cp.client_id_ref
  where name='Rahul';

- **OUTPUT: -**

Explain Notifications Messages Data Output

	client_id integer	name character varying (30)	email character varying (30)	reg_date date	exp_date date	sub_policies_id_ref smallint	agent_id_ref smallint
1	1007	Rahul	rh@gmail.com	2020-06-10	2023-06-10	9002	5101

12. List all the clients (id, name) whose policies will expire in 2024 and the agent's ID 5104.

- SELECT c.client_id,c.name,cp.exp_date,a.agent_id FROM client AS c JOIN client_policies AS cp
  ON c.client_id = cp.client_id_ref JOIN agents AS a
  ON a.agent_id = cp.agent_id_ref
  WHERE cp.exp_date BETWEEN '2024-01-01' AND '2024-12-31';

- **OUTPUT: -**

Explain Notifications Messages Data Output

	client_id integer	name character varying (30)	exp_date date	agent_id smallint
1	1015	Bhuvneshwar	2024-03-08	5102
2	1013	Sania	2024-09-01	5104
3	1010	Mithali	2024-01-01	5105

13. List out all the policies sold in the year 2019.

- select code,name,cp.reg_date,cp.exp_date from policies as p join sub_policies as sp on p.code=sp.policy_type_code_ref
  join client_policies as cp on sp.sub_policies_id=cp.sub_policies_id_ref
  join policy_payment_details as ppd on
  ppd.client_policy_id_ref=cp.client_policy_id
  where date_of_payment between '2019-01-01' and '2019-12-31' ;

- **OUTPUT: -**

Explain Notifications Messages Data Output

	code integer	name character varying (20)	reg_date date	exp_date date
1	1001	Health Insurance	2019-02-12	2020-02-12
2	1001	Health Insurance	2019-08-21	2022-08-21
3	1003	Vehicle Insurance	2019-07-07	2020-07-07
4	1001	Health Insurance	2019-05-11	2020-05-11

14. Update the city and state of client name ‘Sharma’ to city Gandhinagar and state Gujarat.

- UPDATE client SET
  city = 'Gandhinagar', state = 'Gujarat'
  WHERE name ='Sharma'
  RETURNING *;

- **OUTPUT: -**

Explain Notifications Messages Data Output

	client_id [PK] integer	name character varying (30)	email character varying (30)	dob date	city character varying (20)	state character varying (20)	pincode integer	bank_acc_no numeric (16)
1	1002	Sharma	sharmaji@gmail.com	1966-06-06	Gandhinagar	Gujarat	302001	7032883964450002

15. **List out all the agent ID and client ID who has taken policy between 2020-2021.**

- select client_id,agent_id from client as c join client_policies as cp on c.client_id=cp.client_id_ref
  join agents as a on a.agent_id=cp.agent_id_ref
  where reg_date between '2020-01-01' and '2021-12-31';

- **OUTPUT: -**

Explain Notifications Messages Data Output

	client_id integer	agent_id smallint
1	1008	5101
2	1007	5101
3	1015	5102
4	1003	5102
5	1006	5103
6	1001	5103
7	1013	5104
8	1010	5105
9	1006	5105

16.**List out all clients whose medical expenses are above 2 lakhs.**

- SELECT c.client_id,c.name,hcp.medical_expences FROM client AS c
  JOIN health_claim_policy AS hcp
  ON c.client_id = hcp.client_id_ref
  WHERE hcp.medical_expences >= 200000;

- **OUTPUT: -**

Explain Notifications Messages Data Output

	client_id integer	name character varying (30)	medical_expences double precision
1	1003	Virat	350000
2	1006	Mahendra	200000

Summarization

- An insurance agency is a Financial Agency that provides different types of insurance policies to protect individuals and businesses against the risk of financial losses in return for regular payments of premiums.

- In this project, we have created one database management system which mainly focuses on three kinds of policies, i.e., Health (individual, corona kavach), Vehicle (car, bike), Property (commercial, private) insurance. Our agents connect us with the clients.

- We require client details (health info, vehicle info, property info) to proceed further. After the client has purchased a policy, the individual must pay a premium (policy payment details).

- In case of financial losses/ hospitalization, the individual can claim policy (health claim policy, vehicle claim policy, property claim policy) at any time before the expiry date of the policy period that had purchased.