The Problem

Client says: *"I'm 45, married to Sarah, we've got two kids — Lily who's 12 and Jack who's 8 — and I'm a finance director at Barclays, been there 15 years."*

That single response answers **9 questions** across 3 sections:

- Age/DOB (Section 1)
- Marital status (Section 1)
- Partner name (Section 2)
- Number of children (Section 2)
- Children's names and ages (Section 2)
- Occupation (Section 3)
- Employer (Section 3)
- Industry (Section 3)
- Length of service (Section 3)

You absolutely cannot then ask "What's your marital status?" or "Do you have children?"

Solution Architecture

1. Entity Extraction Layer

Every response runs through NLP extraction before the next question is selected:

```
INPUT: "I'm 45, married to Sarah, we've got two kids —
        Lily who's 12 and Jack who's 8"

EXTRACTED ENTITIES:
├── client_age: 45
├── marital_status: married
├── partner_name: Sarah
├── children_count: 2
├── child_1_name: Lily
├── child_1_age: 12
├── child_2_name: Jack
└── child_2_age: 8

CONFIDENCE SCORES:
├── client_age: 0.95 (explicit)
├── marital_status: 0.98 (explicit)
├── partner_name: 0.90 (implied spouse)
└── children_dependency: 0.70 (inferred from age, needs confirmation)
```

2. Slot-Filling Status Tracker

Maintain a real-time status for every data field:

Status	Meaning	Action
`empty`	No data captured	Ask the question
`inferred`	Extracted from another answer, < 85% confidence	Confirm before moving on
`captured`	Extracted with high confidence or directly answered	Skip the question
`confirmed`	Client explicitly confirmed	Locked — don't revisit
`conflicting`	New answer contradicts previous	Flag and clarify
`declined`	Client refused to answer	Note and move on

3. Dynamic Question Selection

The conversation engine doesn't follow a fixed script. It:

1. Checks which slots are still `empty` or `inferred`
2. Selects the next highest-priority unfilled question
3. Skips anything already `captured` or `confirmed`
4. Groups related confirmations together

Practical Conversation Patterns

Pattern A: Skip Entirely (High Confidence)

Client said: "We own our home outright — paid off the mortgage five years ago."

System extracts:

- `property_ownership: yes` (0.98)
- `mortgage_status: none` (0.95)
- `mortgage_balance: 0` (0.95)

System skips: All 8 mortgage questions (lender, balance, rate, term, payment, type, ERCs, repayment method)

Next question: Moves to other properties or liabilities

Pattern B: Confirm and Extend (Medium Confidence)

Client said: "I've got a couple of pensions from old jobs, not sure exactly what's in them."

System extracts:

- `dc_pensions_exist: yes` (0.90)
- `pension_count: ≥2` (0.75)
- `pension_values: unknown` (explicit)

System responds:

> "Got it — you've got pensions from previous employers. Do you know roughly how many, and who they're with? If not, we can request the details later."

Why: Enough to know pensions exist, but needs clarification on details.

Pattern C: Implicit Confirmation (Weave Into Next Question)

Client said: "I'm a partner at a law firm."

System extracts:

- `employment_status: self-employed/partner` (0.85)
- `occupation: solicitor/lawyer` (0.80)
- `industry: legal` (0.95)

Instead of asking: "What's your employment status?"

System asks:

> "As a partner, how do you take your income — salary, drawings, dividends, or a mix?"

Why: Confirms employment status implicitly while gathering new information. Feels natural, not robotic.

Pattern D: Conflict Resolution

Earlier: Client said household income is £80,000

Later: Client says "My salary is £95,000 and my wife earns about £60,000"

System detects: Conflict — earlier total (£80k) ≠ sum of stated incomes (£155k)

System responds:

> "Just to make sure I've got this right — you mentioned earlier your household income was around £80,000, but it sounds like together you're earning closer to £155,000. Which is more accurate?"

Why: Don't silently overwrite. Surface the conflict, get clarification.

Pattern E: Partial Answer Completion

Client said: "We've got two kids in private school."

System extracts:

- `children_count: 2` (0.95)
- `private_education: yes` (0.98)
- `children_names: empty`
- `children_ages: empty`
- `school_fees: empty`

System responds:

> "Two in private school — that's a significant commitment. What are their ages, and roughly what are the annual fees?"

Why: Acknowledges what they said, fills the gaps naturally, doesn't ask "Do you have children?" like an idiot.

Implementation: Question Selection Logic

```
FUNCTION select_next_question():

    # 1. Get all required fields for current section
    required_fields = get_section_fields(current_section)

    # 2. Filter to unfilled or low-confidence
    pending = [f for f in required_fields
               if f.status in ('empty', 'inferred')
               and f.confidence < 0.85]

    # 3. If pending fields can be grouped, combine
    IF can_group(pending):
       RETURN grouped_confirmation_question(pending)
```

```
# 4. If all fields filled, confirm and move on
IF len(pending) == 0:
    IF section_needs_confirmation(current_section):
        RETURN section_summary_confirmation()
    ELSE:
        RETURN move_to_next_section()

# 5. Otherwise, ask highest priority pending question
RETURN get_question(highest_priority(pending))
```

Confirmation Strategies

Micro-confirmations (During Flow)

Weave confirmation into the conversation:

> "So you're 45, married to Sarah, with Lily and Jack still at home — and you're planning to work until around 60. Does that sound right?"

Use when: Transitioning between sections. Confirms 4-5 facts without sounding like a checklist.

Section Summaries (At Transitions)

At the end of major sections:

> "Let me make sure I've got your income picture right: £95k salary, potential £20k bonus, plus Sarah's £60k — so around £155-175k household income depending on the bonus. Anything I've missed?"

Use when: Complex sections (income, assets) where accuracy is critical.

End-of-Call Summary

> "Before we finish, let me read back the key points..."

Use when: Closing the conversation. Required for FCA audit trail.

Edge Cases to Handle

Scenario	Response
Client answers question not yet asked	Extract and slot-fill, acknowledge: "Good to know — I'll note that down"
Client contradicts themselves	Surface gently: "Earlier you mentioned X, but now it sounds like Y — which is right?"
Client says "same as before"	Link to previous answer: "So same as [partner's name] — [previous value]?"
Client gives range	Capture range, ask for best estimate: "So somewhere between £50-70k — what would you say is typical?"
Client says "I don't know"	Mark as `unknown`, note to follow up: "No problem — we can get that from your statement"
Client declines to answer	Mark as `declined`, don't push: "That's fine — we can come back to that if needed"

Data Schema Implication

Your backend needs:

```
{
  "field_id": "client_annual_salary",
  "value": 95000,
  "currency": "GBP",
  "status": "captured",
  "confidence": 0.95,
  "source": "response_to_question_4.1.1",
  "timestamp": "2026-01-24T14:32:00Z",
  "verbatim": "My salary is £95,000",
  "confirmed": true,
  "confirmed_at": "2026-01-24T14:45:00Z"
}
```

Every field tracks:

- **Value** — what was captured
- **Status** — empty/inferred/captured/confirmed/declined
- **Confidence** — NLP extraction certainty
- **Source** — which question or utterance it came from
- **Verbatim** — exact client words (for soft facts)
- **Confirmation** — whether explicitly verified

Summary

Principle	Implementation
Never ask what you already know	Slot-filling with status tracking
Confirm inferences naturally	Weave into follow-up questions
Handle conflicts transparently	Surface and ask, don't silently overwrite
Group related confirmations	Section summaries at transitions
Track provenance	Every field knows where it came from