```
        import json
        import zoneinfo
        from datetime import datetime

        if date_raw.lower() == "latest":
            return "latest", None

        # 現在日時（JSTなど）を明示して相対日時を解釈させやすくする
        tz = zoneinfo.ZoneInfo(self.timezone)
        now = datetime.now(tz)
        today_str = now.strftime("%Y-%m-%d")
        weekday = ["Mon", "Tue", "Wed", "Thu",
                   "Fri", "Sat", "Sun"][now.weekday()]

        # モデルが「出力」を引数として埋める function schema
        tools = [
            {
                "type": "function",
                "function": {
                    "name": "resolve_date_for_frankfurter",
                    "description": (
                        "Choose a SINGLE calendar day for "
                        "the Frankfurter API. "
                        "Return it in ISO 'YYYY-MM-DD'. Only return 'latest' "
                        "when the text explicitly means "
                        "the most recent available rate or "
                        "when the date truly cannot be determined. "
                        "Prefer a concrete past calendar day when possible. "
                        "Honor the provided timezone and 'today'."
                    ),
                    "parameters": {
                        "type": "object",
                        "properties": {
                            "normalized_date": {
                                "type": "string",
                                "description":
                                "The resolved date for Frankfurter "
                                "in 'YYYY-MM-DD' or 'latest'."
                            },
                            "confidence": {
                                "type": "number",
                                "description":
                                "0.0-1.0 subjective confidence of the mapping",
                                "minimum": 0.0, "maximum": 1.0
                            },
                            "policy_applied": {
                                "type": "string",
                                "description":
                                "A short note of the disambiguation rule "
                                "you applied."
                            }
                        },
                        "required": ["normalized_date"]
                    },
                },
            }
        ]

        # 強い指示＋少数ショット例で「latest」濫用を抑制
        system = (
            "You are a precise date normalizer for currency queries.¥n"
            "Rules:¥n"
            "1) Output exactly one calendar day in 'YYYY-MM-DD'.¥n"
            "2) Only return 'latest' when the user literally means 'latest' "
            "or the date cannot be determined.¥n"
            "3) Prefer a concrete, most likely intended past date if text is "
            "relative (e.g., 'last Friday').¥n"
            "4) Respect the timezone and 'today' provided.¥n"
            "5) If the text is a month/quarter/range, pick the most plausible "
            "single representative business day "
            "(e.g., the last calendar day of that period; if weekend/holiday "
            "ambiguity, pick the nearest prior weekday)."
        )

        few_shot_examples = (
            f"Today (timezone={self.timezone}) is {today_str} ({weekday}). "
            "Examples:¥n"
            "- 'last Friday' -> pick the Friday of the previous week "
            "in this timezone.¥n"
            "- '2024/6/1' or 'June 1, 2024' -> 2024-06-01.¥n"
            "- '2024-06' -> choose 2024-06-30 (or nearest prior weekday "
            "if rule 5 applies).¥n"
            "- 'end of last month' -> choose the last calendar day of "
            "the previous month.¥n"
            "- 'quarter 2 of 2024' -> choose 2024-06-30 "
            "(or prior weekday if needed).¥n"
            "- 'latest' -> latest.¥n"
            "Return via the function call ONLY."
```

```
        )

        user = (
            "Normalize this free-form date text for Frankfurter.¥n"
            f"Timezone: {self.timezone}¥n"
            f"Today: {today_str} ({weekday})¥n"
            f"Text: {date_raw}"
        )

        messages = [
            {"role": "system", "content": system},
            {"role": "system", "content": few_shot_examples},
            {"role": "user", "content": user},
        ]

        resp = self.client.chat.completions.create(
            model=self.model,
            messages=messages,
            temperature=0,
            tools=tools,
            tool_choice={"type": "function",
                         "function": {"name": "resolve_date_for_frankfurter"}},
        )
```