Problem Statement: Expense Policy Rules Engine

You are tasked with designing a system that evaluates **business expenses** submitted by employees.

Managers can define **policies (rules)** to control expenses, ensuring employees do not misuse corporate cards or exceed allowances.

Your goal is to build a rules engine that:

1. Evaluates individual expenses against a set of rules.
2. Evaluates aggregated trip-level expenses against a set of rules.
3. Flags any violations clearly.

Input Format

- A list of **expenses**, where each expense is represented as a dictionary/map of string keys and values.
  Example keys:
  - `"expense_id"`
  - `"trip_id"`
  - `"amount_usd"`
  - `"expense_type"` (e.g., `"restaurant"`, `"airfare"`, `"entertainment"`)
  - `"vendor_type"`

    - `"vendor_name"`

- A list of **rules** to evaluate. Rules can be applied at:
    - **Expense level** (per individual expense).
    - **Trip level** (across all expenses belonging to a trip).

Basic Rules

Start with the following rules:

1. **No restaurant expense can exceed $75.**
2. **No airfare expenses are allowed.**
3. **No entertainment expenses are allowed.**
4. **No single expense can exceed $250.**

Extended Rules

Later, add support for **trip-level rules** such as:

5. **A trip cannot exceed $2000 in total expenses.**
6. **Total meal (restaurant) expenses per trip cannot exceed $1000.**

Output Format

- For each **expense**, return whether it is **APPROVED** or **REJECTED**, with reasons for rejection.
- For each **trip**, return whether it is **OK** or has **VIOLATIONS**, with reasons.

Example Input

```
expenses = [
  {
    "expense_id": "001",
    "trip_id": "trip1",
    "amount_usd": "80",
    "expense_type": "restaurant",
    "vendor_name": "Outback Roadhouse"
  },
  {
    "expense_id": "002",
    "trip_id": "trip1",
    "amount_usd": "120",
    "expense_type": "supplies",
    "vendor_name": "Staples"
  },
  {
    "expense_id": "003",
    "trip_id": "trip1",
    "amount_usd": "199",
    "expense_type": "airfare",
    "vendor_name": "Delta Airlines"
  },
  {
    "expense_id": "004",
    "trip_id": "trip1",
    "amount_usd": "260",
    "expense_type": "hotel",
    "vendor_name": "Marriott"
  },
  {
```

```
    "expense_id": "005",
    "trip_id": "trip1",
    "amount_usd": "70",
    "expense_type": "restaurant",
    "vendor_name": "Chipotle"
  },
  {
    "expense_id": "006",
    "trip_id": "trip1",
    "amount_usd": "40",
    "expense_type": "entertainment",
    "vendor_name": "AMC Theaters"
  }
]
```

Expected Output

```
Expense Violations:
001 -> REJECTED: [Restaurant expense exceeds $75]
002 -> APPROVED
003 -> REJECTED: [Airfare expenses not allowed]
004 -> REJECTED: [Expense exceeds $250 limit]
005 -> APPROVED
006 -> REJECTED: [Entertainment expenses not allowed]

Trip Violations:
trip1 -> VIOLATIONS: [Total trip expenses $769 exceed $2000? NO, so OK he
                      But total meal expenses $150 exceed $1000? NO, so (
```

(If the trip total crossed $2000 or meal total $1000, those rules would trigger.)

Additional Notes

- The system should be **extensible**: managers may eventually have **hundreds of rules**, so adding rules should not require rewriting the core evaluation logic.
- The same framework should support **future custom rules** (e.g., weekend-only expenses, vendor blacklists, monthly budget caps).