Laozi DSL by Pi.Tech

Laozi Expression Language (DSL) Reference

The Laozi Expression Language is a domain-specific language for expressing analytical computations on structured data. It allows you to write human-readable expressions that compile to optimized SQL queries while maintaining strict type safety and constraint enforcement.

> *Expressions are deterministic — the same input always produces the same result. This is core to Laozi's anti-hallucination philosophy.*

Operator Precedence (PEMDAS)

The DSL follows mathematical precedence rules:

Priority	Operator	Description	Example
1 (Highest)	`()`	Parentheses / Function calls	`SUM(amount)`
2	`^`	Exponentiation	`value ^ 2`
3	`* /`	Multiplication, Division	`total / count`
4 (Lowest)	`+ -`	Addition, Subtraction	`income - expenses`

Function Categories

Aggregation Functions

Function	Description	Example
`SUM(field)`	Sum of values	`SUM(amount)`

`AVG(field)`	Mean of values	`AVG(amount)`
`COUNT(*)`	Count of records	`COUNT(*)`
`MIN(field)`	Minimum value	`MIN(amount)`
`MAX(field)`	Maximum value	`MAX(amount)`

Math Functions

Function	Description	Example
`ROUND(expr, n)`	Round to n decimals	`ROUND(ratio, 2)`
`ABS(expr)`	Absolute value	`ABS(balance)`
`NULLIF(a, b)`	Returns null if a = b	`NULLIF(total, 0)`
`SQRT(expr)`	Square root	`SQRT(variance)`

Statistical / Forensic Functions

Function	Description	Example
`GINI(field GROUP_BY(key))`	Gini coefficient for concentration analysis	`GINI(amount GROUP_BY(payee))`
`STDEV(field)`	Standard deviation	`STDEV(amount)`
`CHANGE(field, period)`	% change over period	`CHANGE(revenue, 3 months)`
`BENFORD(field)`	Benford's Law digit analysis	`BENFORD(amount)`

Time & Period Functions

Function	Description	Example
`OVER(n unit)`	Rolling window	`SUM(x) OVER(30 days)`
`PERIOD(name)`	Named period	`SUM(x) PERIOD(YTD)`

`ON(pattern)`	Day-of-week filter	`WHERE(ON(weekends))`

Named Periods

Period	Definition
`YTD`	Year to date (Jan 1 → today)
`MTD`	Month to date (1st → today)
`QTD`	Quarter to date
`last_year`	Full previous calendar year
`last_quarter`	Full previous quarter
`last_month`	Full previous month

Conditional & Filter Functions

Function	Description	Example
`WHERE(condition)`	Filter records	`SUM(x) WHERE(type='credit')`
`AND / OR`	Boolean connectives	`WHERE(a > 5 AND b < 10)`
`GROUP_BY(field)`	Grouping for aggregation	`GINI(x GROUP_BY(y))`

Expression Examples

Income-to-Expense Ratio

Calculate the income-to-expense ratio as a percentage over the last 30 days, rounded to 2 decimal places.

```
ROUND(
  (SUM(amount) WHERE(type = 'income') OVER(30 days))
  /
  NULLIF(SUM(amount) WHERE(type = 'expense') OVER(30 days), 0)
  * 100,
```

```
  2
)
```

Credit Transaction Count

Count all credit transactions in the current year.

```
COUNT(*) WHERE(type = 'CREDIT') PERIOD(YTD)
```

Payee Concentration Index

Calculate the Gini coefficient for payment distribution across payees to detect concentration risk.

```
GINI(amount GROUP_BY(payee))
```

Quarterly Revenue Change

Measure the percentage change in total revenue over the last 3 months.

```
CHANGE(amount, 3 months)
```

High-Value Transaction Detection

Sum amounts of transactions exceeding $10,000 on weekends over the last 90 days.

```
SUM(amount) WHERE(amount > 10000 AND ON(weekends)) OVER(90 days)
```

Expense Volatility

Calculate the standard deviation of monthly expenses to measure spending volatility.

```
STDEV(amount) WHERE(type = 'expense') OVER(last_year)
```

How DSL Prevents Hallucination

The DSL is a key component of Laozi's anti-hallucination strategy:

1. **Deterministic Execution** — Expressions compile to SQL and execute against real data, not LLM inference.
2. **Type Safety** — Functions enforce input types; aggregation functions require numeric fields.
3. **Pre-computed Results** — The LLM receives the computed result, not the expression. It narrates, never calculates.
4. **Audit Trail** — Every expression is logged with its compiled SQL, input data, and output for full traceability.

Integration with Laozi Engine

DSL expressions are defined within category thresholds and are evaluated before the LLM is invoked:

```
// In Go
engine.AddCategory(laozi.Category{
    ID: "revenue",
    Thresholds: []laozi.Threshold{{
        Metric: "monthly_revenue",
        Expression: `SUM(amount) WHERE(type = 'revenue') OVER(30 days)`,
        Min: 10000,
        Max: 50000,
    }},
})

// In Java
engine.addCategory(new Category("revenue", "Revenue", List.of(
    new Threshold("monthly_revenue",
        "SUM(amount) WHERE(type = 'revenue') OVER(30 days)",
        10000, 50000, "USD", "Benchmark")
)));
```