Adaptive Risk Scoring Queries for Longitudinal Student Mental Health Data

COMP-8157: Advanced Database Topics

Hemit Rana Sarvesh Solanke Charmiben Patel Chetan Thakur
Jasmeen Kaur Harpreet Singh Anurag Sharma

School of Computer Science
University of Windsor
Windsor, ON, Canada

Winter 2026

Project Overview

Context

Universities deploy longitudinal mental health surveys to track student well-being across 20+ indicators over 2–4 cycles per year.

The Challenge

Risk models evolve continuously — weights change, factors are added/removed, thresholds are adjusted — yet database systems assume **static query semantics**.

Our Goal

Design and evaluate execution strategies for evolving analytical queries, providing empirical guidance on which approach minimizes cost under different change patterns.

Motivation

The Problem

- When definitions change, recomputing all risk scores can take **10–100× longer** than static semantics
- Sleep quality added as factor (*structural change*)
- Stress threshold adjusted (*parametric change*)
- Vector embeddings integrated (*hybrid change*)

The Research Gap

- Database research assumes **static semantics**
- Modern optimizers don't model query evolution
- No guidance on strategy selection under semantic change
- Critical blind spot for healthcare, finance, policy systems

Problem Statement

Risk-Scoring Query Definition

A risk-scoring query is parameterized by definition D specifying:

1. Set of factors F (stress, sleep, performance)
2. Weights w (e.g., $w_{\text{stress}} = 0.4$)
3. Aggregation granularity (rolling 12-week window)
4. Thresholds (score $>$ 75th percentile)
5. Vector similarity thresholds for text-derived signals

Three Categories of Semantic Change:

PARAMETRIC	STRUCTURAL	HYBRID
Adjust weights, thresholds, or similarity cutoffs	Add/remove factors, change aggregation or temporal	Couple structured and vector-based signals

Research Question

Central Question

What execution strategy minimizes **total cost**

(latency + storage + maintenance)

of computing risk scores under **evolving definitions**?

Key Consideration

Does the answer depend on frequency, type, and magnitude of change?

- Change frequency: weekly vs. monthly vs. quarterly
- Change type: parametric vs. structural vs. hybrid
- Dataset size: number of students and survey cycles

Related Work

Materialized Views

Gupta & Mumick, Nikolic et al.

Assumes fixed view definition

Optimizes data refresh, not query evolution

Adaptive Query Processing

STREAM, Babu & Widom, Psaroudakis

Adapts to variable data characteristics

Not to evolving query semantics

Learned Optimizers

Bao, Kersten et al.

Predicts optimal plans for static queries

Training assumes fixed logical structure

Polyglot Databases

Dong et al., hybrid systems

Optimizes static hybrid definitions

Doesn't address co-evolution

Novelty & Contributions

Breaking the Assumption

Analytical queries are **NOT** semantically static once deployed

1. Semantic Change Model

Formal categorization: parametric, structural, hybrid changes with annotations

2. Similarity-Aware Reuse

Detect unchanged subexpressions across query versions to avoid recomputation

3. Polyglot Evaluation

Measure how text-derived vectors co-evolve with structured risk definitions

4. Empirical Guidance

Evidence of which strategy dominates under different change patterns

Actionable insights for practitioners & database designers building future optimizers

System Design

Technology Stack

- **PostgreSQL 14+** with pgvector extension
- **Python 3.9+** pandas, sqlalchemy
- **sentence-transformers** for embeddings
- **Standard SQL & libraries** (no custom engine)

Four Execution Strategies

1. **SQL Recomputation**
   Fresh query per change
2. **Materialized Views**
   Refresh affected views
3. **Incremental Computation**
   Recompute only affected parts
4. **Window-Function-Centric**
   Minimize time-based redundancy

Data Model & Dataset

Database Schema (6 Tables)

1. **students** — demographic info
2. **survey_cycles** — temporal metadata
3. **questions** — survey items
4. **structured_responses** — numeric scores
5. **vector_responses** — text embeddings (pgvector)
6. **risk_definitions** — JSON model specifications

DATASET

Source:

American College Student Health Survey (ACSHS)

Scale:

5,000–10,000 students
6–8 survey cycles

Content:

- Structured: stress, sleep, academic scores
- Unstructured: free-text stress descriptions

Implementation Plan

System Components

1. PostgreSQL schema setup (SQL DDL)
2. Data ingest pipeline (pandas)
3. Vector embedding generation (sentence-transformers)
4. Four strategy implementations (SQL procedures + Python wrappers)
5. DSL compiler (Python)
6. Evaluation harness (timing loops, CSV logging)

Team Allocation (7 members)

- **Schema & Data**
  2 people
- **Baseline & Views**
  2 people
- **Incremental & Windows**
  2 people
- **DSL & Evaluation**
  1 person

Project Timeline

Duration	Milestone
Weeks 1–2	**Schema & Data**
	Ingest, embeddings, validation
Weeks 3–4	**Baseline & Views**
	SQL recomputation, materialized views
Weeks 5–6	**Incremental & Windows**
Weeks 7–8	**DSL & Evaluation**
	Compiler, change traces, full evaluation
Weeks 9–10	**Analysis & Documentation**
	Results, plots, final report

Risk Mitigation Strategies

Risk 1: Data Availability

Mitigation: Pre-download public datasets (UMN SRCD, APA). If unavailable, generate synthetic data with realistic correlation structures.

Risk 2: Vector Embedding Cost

Mitigation: Generate embeddings offline once. Parallelize with multiprocessing. Reduce to 100–150 dimensions if needed.

Risk 3: Implementation Complexity

Mitigation: Start with simple dependency graph. Iterative testing. If too complex, fall back to 3 of 4 strategies.

Risk 4: Limited Evaluation Time

Mitigation: Prioritize common scenarios (monthly changes, parametric + structural). Limit to 20–30 semantic change events.

Expected Outcomes

Empirical Results

Comparative latency, storage, and maintenance cost data across 4 strategies under varying change frequencies (weekly, monthly, quarterly) and types (parametric, structural, hybrid)

Strategy Selection Rules

Decision framework: When does materialization dominate? When is incremental computation optimal? How does the answer vary with change patterns?

Publication Target

Systems workshop paper or short conference paper at **SIGMOD**, **VLDB**, or **CIDR**

Impact & Significance

1. Practical Systems Contribution

Quantify trade-offs current benchmarks and optimizers don't model

- First empirical comparison of execution strategies under semantic evolution
- Actionable cost models for practitioners
- Strategy selection rules based on change patterns

2. Fills Critical Research Gap

First systematic characterization of semantic evolution workload dimension

- Challenges fundamental assumption of static query semantics
- Extends view maintenance and incremental computation theory

3. Broad Applicability

Insights apply to healthcare, finance, policy analytics — any evolving analytical system

Key Takeaways

1. **Database systems assume static query semantics**
   - Real workloads violate this assumption
   - Risk models, policies, and analytics evolve continuously
2. **We model semantic evolution as a first-class workload characteristic**
   - Parametric, structural, and hybrid change categories
   - Formal change model with annotations
3. **Systematic evaluation provides actionable strategy-selection guidance**
   - Four execution strategies compared empirically
   - Evidence-based decision framework
4. **Results inform future optimizer development**
   - New dimension for cost models and benchmarks
   - Practical insights for system designers

Thank You!

Contact Information

University of Windsor
Windsor, ON, Canada

Team Contact:

rana6c@uwindsor.ca • solanks@uwindsor.ca • patel4mc@uwindsor.ca

thakuc1@uwindsor.ca • kaur8m@uwindsor.ca

singh4x@uwindsor.ca • sharma9p@uwindsor.ca