BRAD is a cloud-native multi-engine database system where “one interface fits most”

The modern cloud data mesh is painful to create, manage, and use

- “One size does not fit all” led to a plethora of specialized cloud services for data workloads
- High administrative burden: select the “best” engines, provision resources, evolve the mesh
- The holy grail: One system with state-of-the-art performance for all workloads while leveraging existing specialized systems

Learned query planning

- Routing: Select the best single engine for a query
- Multi-engine query planning: How to best split a plan across engines
- Powered by learned query run time models (cost models)
  - Cannot rely on physical plan features because BRAD might need predictions for an engine that is not yet running

Joint mesh, query, cost optimization

- Jointly consider query routing, mesh design, data placement
- Optimize for cost under performance ceiling (or vice-versa)
- BRAD goes beyond cloud auto-scaling; workloads can be moved across engines
  - Driven by an ensemble of models: (i) query run time on different hardware and system load, (ii) data scanned, (iii) data transfer times, (iv) cloud instance resize times

Data consistency and auto ETLs/ELTs

- Efficiently achieving strong consistency across engines?
- Session-based freshness guarantees; implemented using tuple multi-versioning, epoch-based logical snapshots
- Users describe transforms (ETLs/ELTs) using a declarative language
- BRAD auto-schedules transforms on the best system(s); using special technologies if able (e.g., AWS Glue, zero-ETL)

BRAD predicts query run time on different hardware configurations with 18%/22% mean absolute percentage test error Aurora/Redshift

BRAD optimizes a data mesh containing Redshift, saving 4x in cost (change instance type while maintaining SLAs)