PRD: [Name of the project]

PROBLEM ALIGNMENT

1. PROBLEM DEFINITION

Describe the problem (or opportunity) you're trying to solve. Why is it important to our users and our business? What insights are you operating on? And if relevant, what problems are you **not** intending to solve?

Consider:

- **User Impact**: How does this affect data engineers, developers, or business users working in Keboola?
- **Business Impact**: Does this drive enterprise customer retention, multi-tenant growth, or PAYG expansion?
- **Strategic Alignment**: How does this support our platform strategy (performance, developer experience, feature completeness)?

---

2. HIGH-LEVEL APPROACH

Describe briefly the approach you're taking to solve this problem. This should be enough for the reader to imagine possible solution directions and get a very rough sense of the scope of this project.

Example: If "The Problem" was poor Data App startup times affecting developer experience, "The Approach" might be implementing incremental image builds and optimizing the deployment pipeline.

---

3. DEFINE SUCCESS METRICS

Identify where the feature impacts the customer journey and business model:

For Enterprise Customers (Platform Adoption & Retention):

- **Drive Adoption** (Does it increase platform usage depth?) → Track: Active flows, transformations run, data apps deployed, component configurations created, storage volume, job frequency

- **Improve Developer Experience** (Does it reduce friction or improve productivity?) → Track: Time to first value, error rates, support ticket volume, feature usage rates, developer satisfaction scores
- **Enhance Retention & Expansion** (Does it create stickiness or unlock upsell?) → Track: Monthly active projects, DAU/MAU ratio, feature adoption across accounts, churn risk indicators, usage growth trends

For Multi-Tenant / PAYG Growth:

- **Registration & Activation** (Does it improve onboarding?) → Track: Sign-up rate, time to first job, onboarding completion rate, trial-to-paid conversion
- **Engagement** (Does it drive continued usage?) → Track: D7/D14/D30 retention, weekly active users, feature engagement rates, jobs per user
- **Monetization** (Does it drive PAYG revenue?) → Track: PAYG revenue, overage charges, upgrade rates, customer LTV

For Platform Performance:

- Track: Job success rates, average job duration, error rates (e.g., 500 errors), platform stability metrics, API response times

Decision Checkpoint:

- If we can't measure it, we shouldn't be doing this.
- ⚠️ **Beware of Trade-Offs**: Could the initiative harm other metrics? (e.g., adding complexity that hurts onboarding while improving power users)

---

4. SET THE BASELINE

Measure the current state before implementing any changes. Link to the relevant dashboard in Keboola's analytics or telemetry system.

Required baseline data:

- Current metric values (with date range)
- Trend direction (improving, declining, stable)
- Sample size and statistical significance considerations
- Segmentation if relevant (e.g., by customer tier, region, or backend - GCP US East 4 vs Azure North Europe)

If baseline data is unavailable, ask:

- Is the potential impact large enough to justify building measurement first?

- What qualitative signals or proxy metrics can we use? (e.g., support tickets, user feedback, competitive benchmarking)
- Can we instrument measurement as part of the project?

Decision Checkpoint: We shouldn't do this if we can't measure the baseline or establish clear success criteria.

SOLUTION ALIGNMENT

5. KEY FEATURES

Give an overview of what we're building. Provide an organized list of features, with priorities if relevant. Discuss what you're **not** building (or saving for a future release).

Break down the initiative into versions with milestones (especially for L and XL initiatives):

V1: Foundation / Quick Wins

- Core functionality or immediate pain point resolution
- Example: SQL Editor basic query execution, Data Apps startup time reduction from 45s to 15s
- Focus on unblocking users or proving value hypothesis

V2: Feature Completeness

- Enhanced functionality that makes the solution production-ready
- Example: SQL Editor multi-tab support, Data Apps storage API integration
- Focus on matching or exceeding existing workflows (e.g., Snowsight parity)

V3: Differentiation & Compound Value

- Advanced capabilities that create competitive advantage
- Example: SQL Editor AI-powered query assistance, Data Apps Conditional Flows integration
- Focus on leverage and ecosystem effects

Decision Checkpoint: "Dare to kill" along the way if:

- Impact is not proven after V1
- Resource investment becomes disproportionate to returns
- User adoption signals don't validate the hypothesis
- Business priorities shift (e.g., enterprise vs PAYG focus changes)