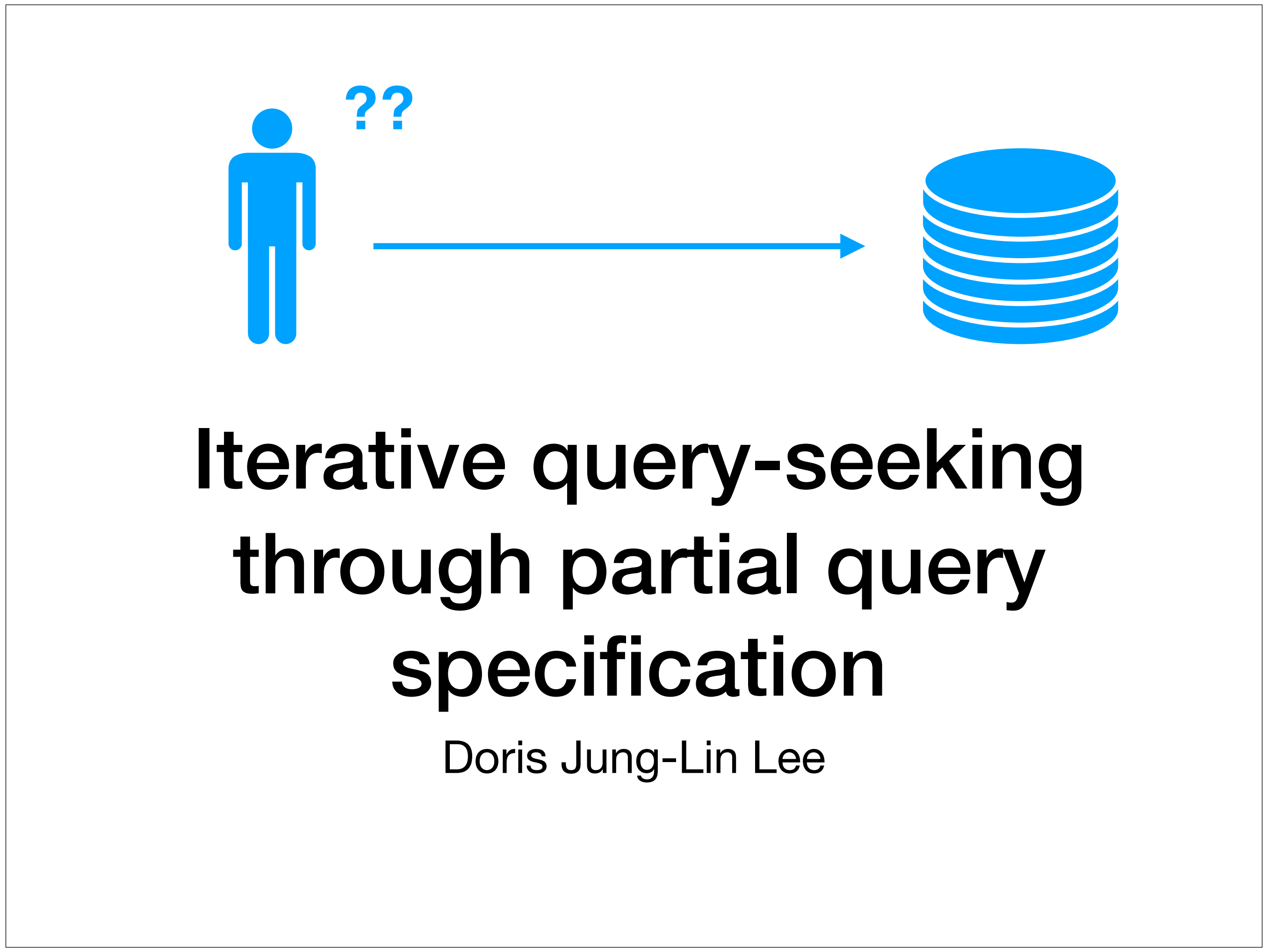


Iterative query-seeking through partial query specification

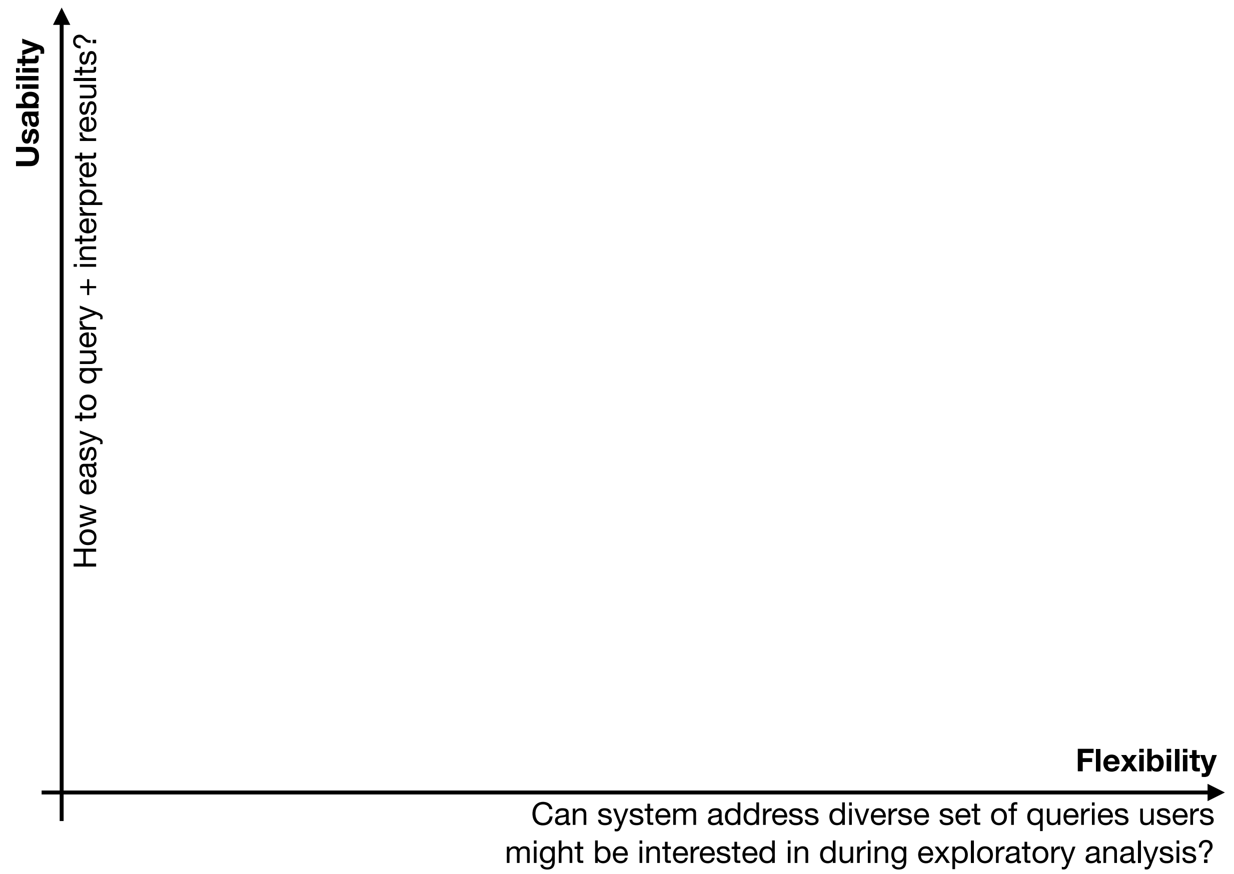
Doris Jung-Lin Lee

Hi everyone, today I'll be talking about my project on designing a new querying language for performing early-stage exploratory data analysis.

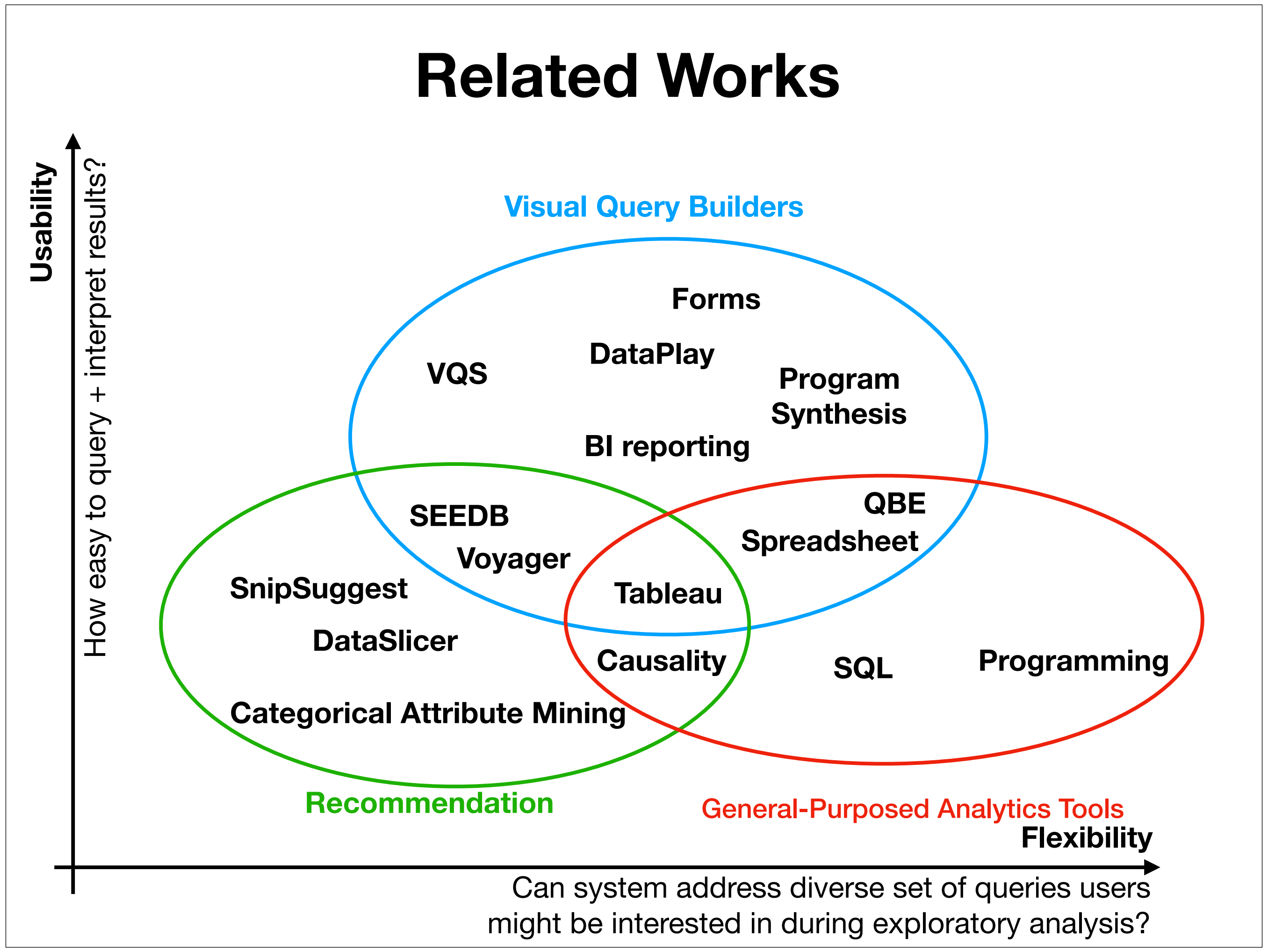
Related Works

My project is motivated by the question: How can we improve the way people query databases?
Over the course of the semester, we've read many papers that looked at this problem. We decompose these systems into two dimensions: flexibility and usability. Usability deals with how easy it is to query and interpret the returned query results, very often this is done through visualizations or system explanations. Flexibility deals with whether the system can handle a diverse set of queries that users might be interested in. The related works in this space all show some tradeoff between these two desired criteria. For example, visual query builders are highly usable for novices, but they are limited in the types of query they can support. General Purposed tools such as Excel and Tableau are very flexible in the queries they support, but there's often too many options so users are often not sure what they want to query and what are the right sequence of steps to reach those objectives. Ideally, we want to be here (*) where we have high usability and flexibility, but currently there is no one-size-fit-all solution to query specification because of this tradeoff.

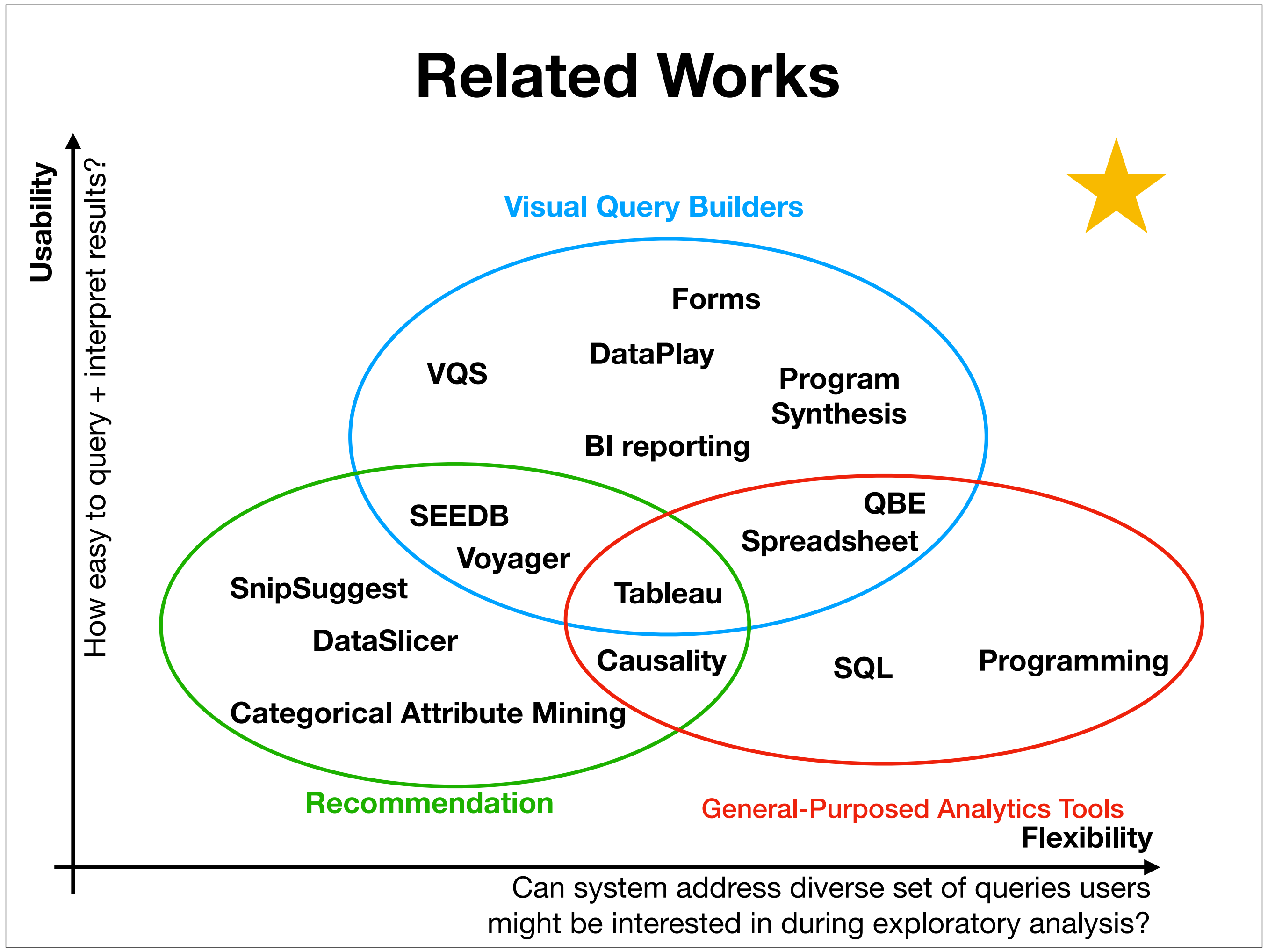
Related Works


Usability
How easy to query + interpret results?
Flexibility
Can system address diverse set of queries users might be interested in during exploratory analysis?

My project is motivated by the question: How can we improve the way people query databases?
Over the course of the semester, we've read many papers that looked at this problem. We decompose these systems into two dimensions: flexibility and usability. Usability deals with how easy it is to query and interpret the returned query results, very often this is done through visualizations or system explanations. Flexibility deals with whether the system can handle a diverse set of queries that users might be interested in. The related works in this space all show some tradeoff between these two desired criteria. For example, visual query builders are highly usable for novices, but they are limited in the types of query they can support. General Purposed tools such as Excel and Tableau are very flexible in the queries they support, but there's often too many options so users are often not sure what they want to query and what are the right sequence of steps to reach those objectives. Ideally, we want to be here (*) where we have high usability and flexibility, but currently there is no one-size-fit-all solution to query specification because of this tradeoff.

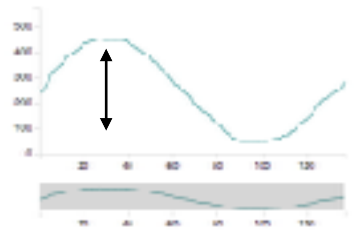

Related Works
Usability
How easy to query + interpret results?
Visual Query Builders
Forms
VQS
DataPlay
Program Synthesis
BI reporting
SEEDB
QBE
Spreadsheet
Voyager
SnipSuggest
Tableau
DataSlicer
Causality
SQL
Programming
Categorical Attribute Mining
Recommendation
General-Purposed Analytics Tools
Flexibility
Can system address diverse set of queries users might be interested in during exploratory analysis?

My project is motivated by the question: How can we improve the way people query databases?
Over the course of the semester, we've read many papers that looked at this problem. We decompose these systems into two dimensions: flexibility and usability. Usability deals with how easy it is to query and interpret the returned query results, very often this is done through visualizations or system explanations. Flexibility deals with whether the system can handle a diverse set of queries that users might be interested in. The related works in this space all show some tradeoff between these two desired criteria. For example, visual query builders are highly usable for novices, but they are limited in the types of query they can support. General Purposed tools such as Excel and Tableau are very flexible in the queries they support, but there's often too many options so users are often not sure what they want to query and what are the right sequence of steps to reach those objectives. Ideally, we want to be here (*) where we have high usability and flexibility, but currently there is no one-size-fit-all solution to query specification because of this tradeoff.

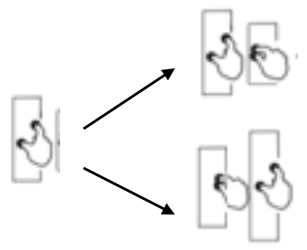

Related Works
Usability
How easy to query + interpret results?
Visual Query Builders
Forms
VQS
DataPlay
Program Synthesis
BI reporting
SEEDB
QBE
Spreadsheet
Voyager
SnipSuggest
Tableau
DataSlicer
Causality
SQL
Programming
Categorical Attribute Mining
Recommendation
General-Purposed Analytics Tools
Flexibility
Can system address diverse set of queries users might be interested in during exploratory analysis?

My project is motivated by the question: How can we improve the way people query databases?
Over the course of the semester, we've read many papers that looked at this problem. We decompose these systems into two dimensions: flexibility and usability. Usability deals with how easy it is to query and interpret the returned query results, very often this is done through visualizations or system explanations. Flexibility deals with whether the system can handle a diverse set of queries that users might be interested in. The related works in this space all show some tradeoff between these two desired criteria. For example, visual query builders are highly usable for novices, but they are limited in the types of query they can support. General Purposed tools such as Excel and Tableau are very flexible in the queries they support, but there's often too many options so users are often not sure what they want to query and what are the right sequence of steps to reach those objectives. Ideally, we want to be here (*) where we have high usability and flexibility, but currently there is no one-size-fit-all solution to query specification because of this tradeoff.

Zenvisage



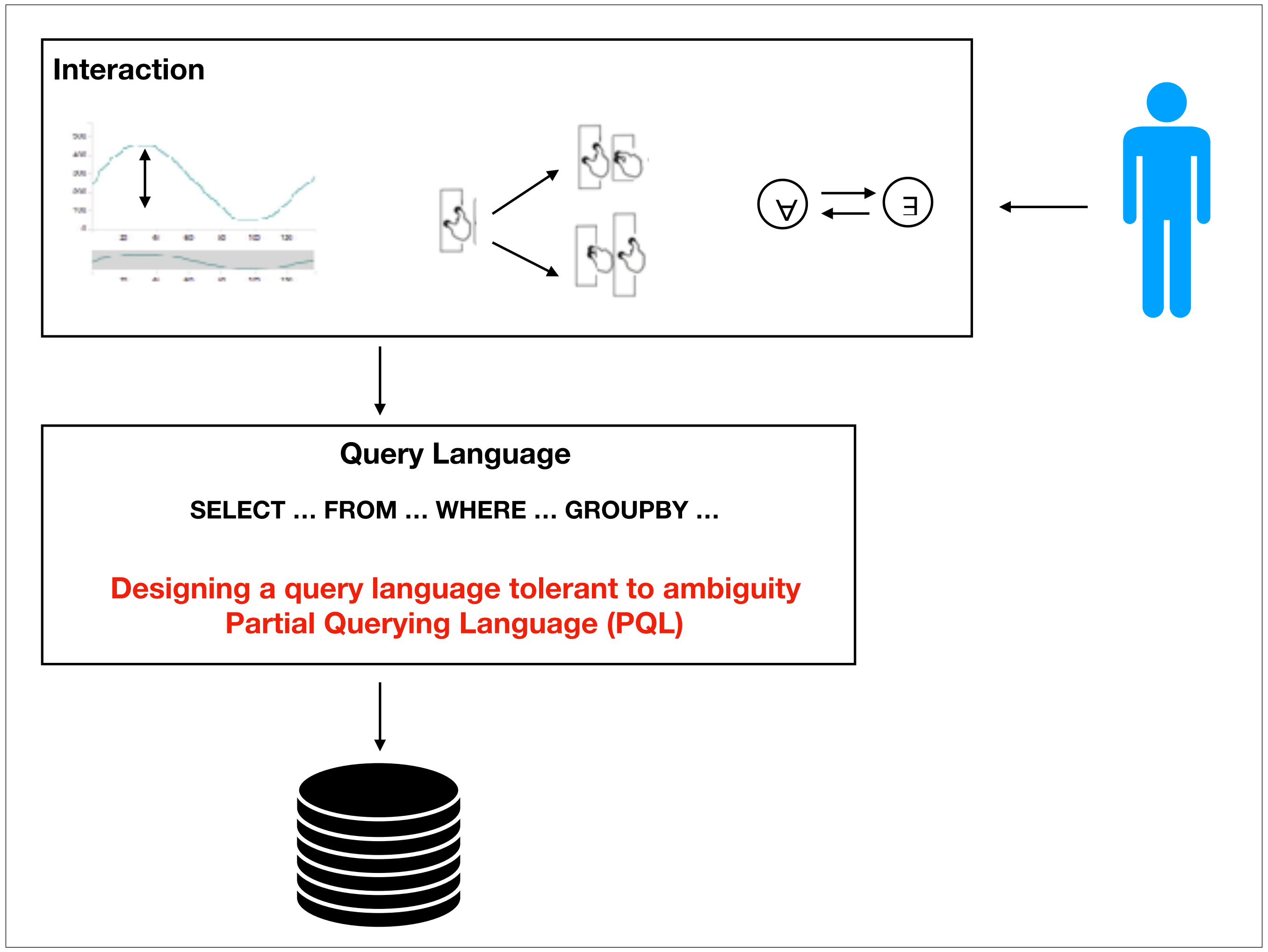
Gesture DB, DB Touch



DataPlay

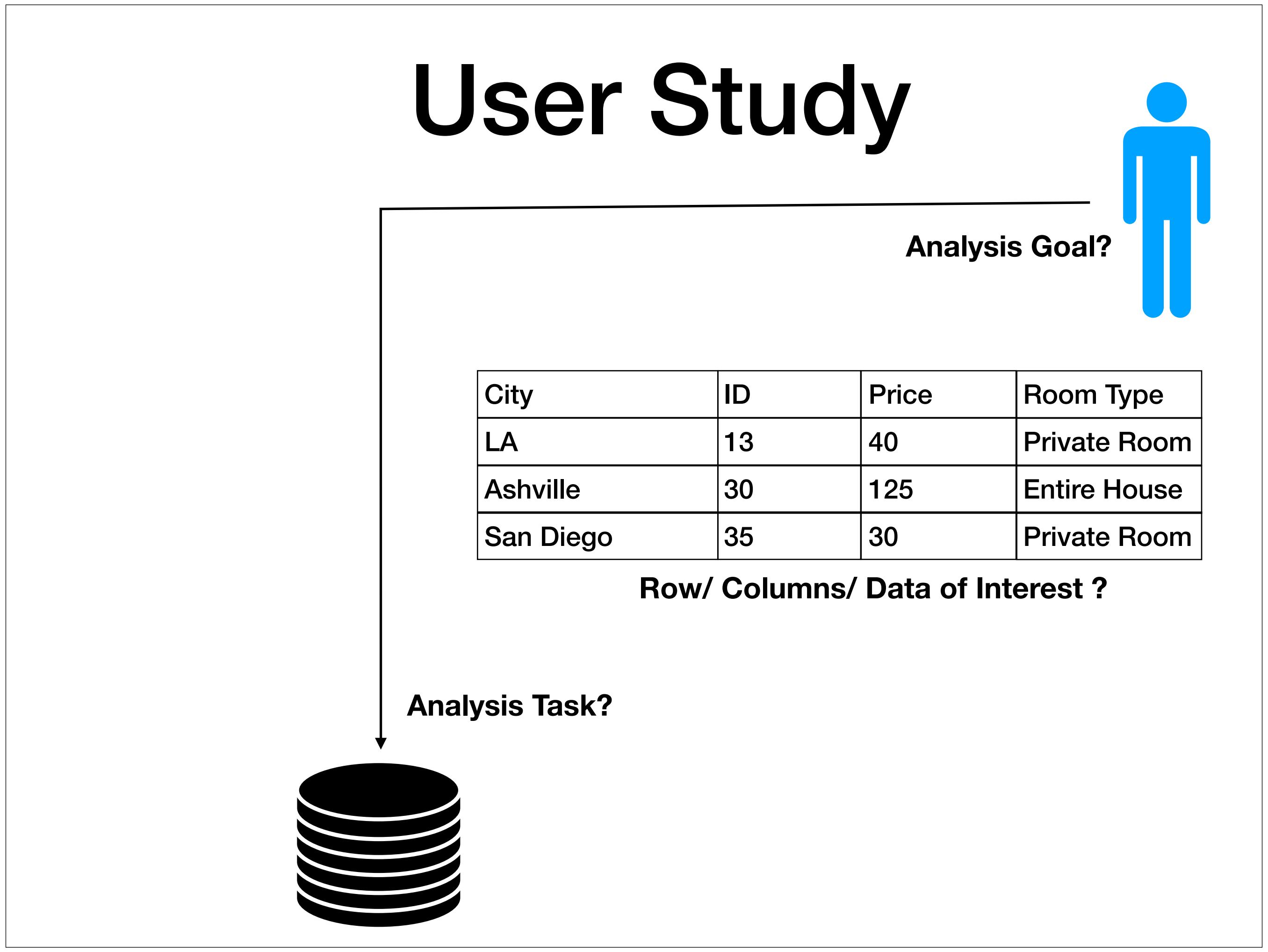
$$\begin{tikzcd}\forall \arrow[r, shift left] & \exists \arrow[l, shift left]\end{tikzcd}$$

I wanted to focus our attention on some of the visual query builders that we looked at in this course. In all of these application, we see a little bit of ambiguity in our query. For example, in Zenvisage a user might first start with a rough sketch, inspect the output, then realize that he doesn’t really care about the magnitude of the y axis as long as the shape is preserved. In DataPlay, a user might start off with a query that ask for students who does well in **any** intro CS courses, but then later realize the question he is really interested in is finding students who does well in **all** CS classes. In gesture-based querying system, they devote a big part of their paper talking about classifying what actions users are trying to do based on the sequence of touch and clicks, since the same sub-sequence gesture can be maps onto different actions. [Being able to perform “semantically similar” query modifications allows users to explore related hypothesis.] The interface-level ambiguity is both a result of the querying mechanism itself and naturally the way that people query data exploratory analysis.


Interaction
∀
∃
Query Language
SELECT … FROM … WHERE … GROUPBY …
Designing a query language tolerant to ambiguity
Partial Querying Language (PQL)

When you abstract away the interactions, these systems are simply sending SQL queries to the database. What if we can have the querying language itself capture some of the ambiguity resulting from the user's intention or interfaces and use it to our advantage to provide query recommendations to users? In this project, I am designing a querying language PQL which is a interface-independent unifying language that can tolerate ambiguity in its specification to generate query recommendations for rapid insight discovery.

Instead of going ahead to build the language and interactions based on what we think the users want, we performed a preliminary user study to figure out a set of specification for PQL. We did this by giving the users a real-world dataset from Airbnb and asking them to brainstorm the types of analysis goals and queries they would be interested in.



User Study

Analysis Goal?

City	ID	Price	Room Type
LA	13	40	Private Room
Ashville	30	125	Entire House
San Diego	35	30	Private Room

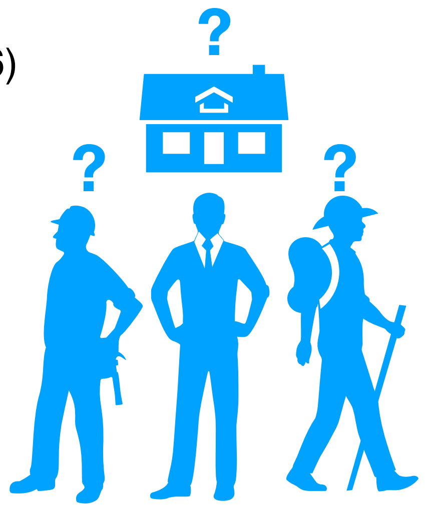
Row/ Columns/ Data of Interest ?

Analysis Task?

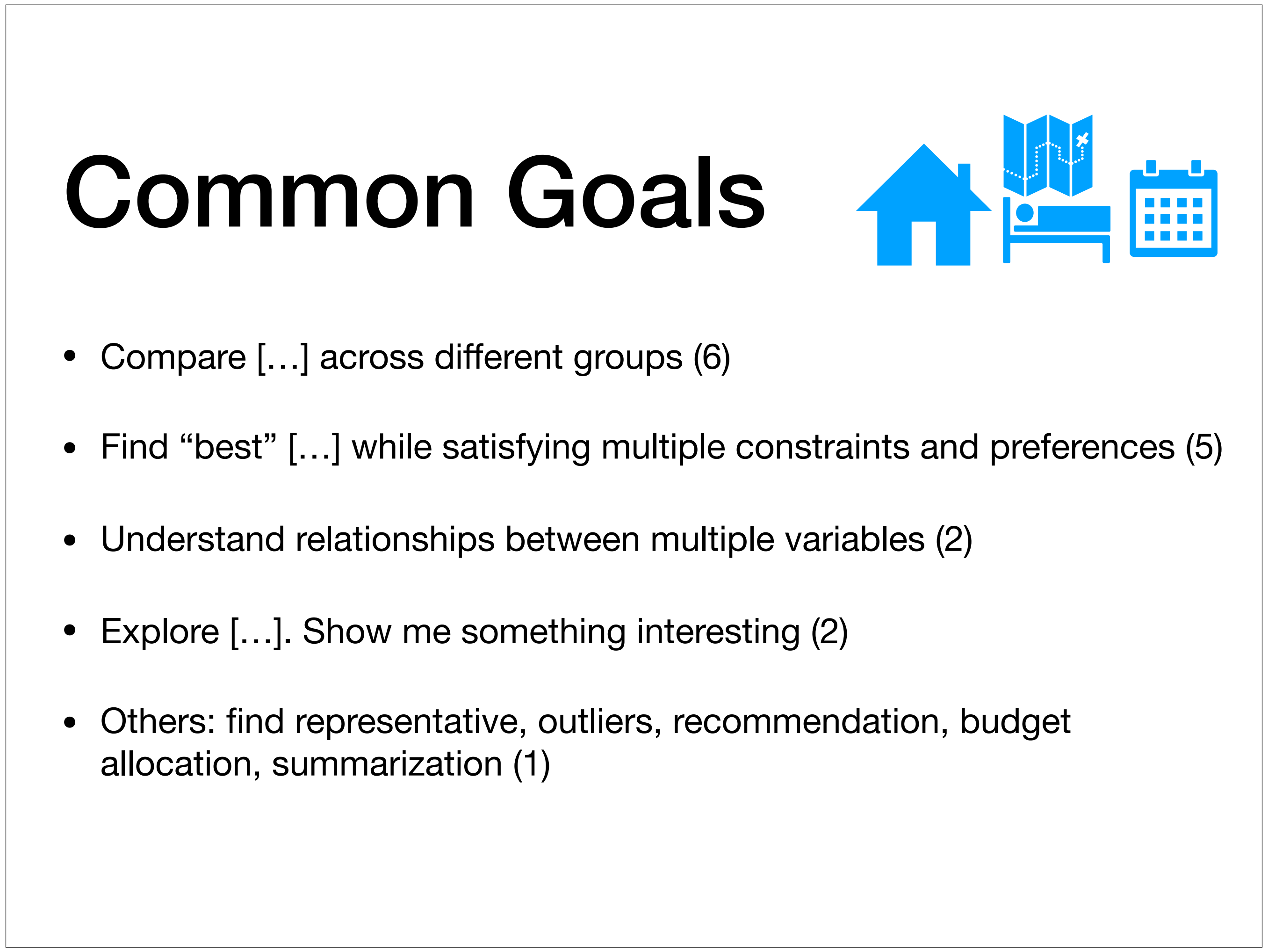
When you abstract away the interactions, these systems are simply sending SQL queries to the database. What if we can have the querying language itself capture some of the ambiguity resulting from the user's intention or interfaces and use it to our advantage to provide query recommendations to users? In this project, I am designing a querying language PQL which is a interface-independent unifying language that can tolerate ambiguity in its specification to generate query recommendations for rapid insight discovery.
Instead of going ahead to build the language and interactions based on what we think the users want, we performed a preliminary user study to figure out a set of specification for PQL. We did this by giving the users a real-world dataset from Airbnb and asking them to brainstorm the types of analysis goals and queries they would be interested in.

(Preliminary) Study Results

- Total 22 scenarios (so far!)
- Task Complexity: 1~4 steps (μ = 2.6; σ=0.76)
- Common tasks:
  - SQL functionalities (group-by, filter, sort, aggregation, top-k)
  - derive new column
  - feature discovery
  - similarity search
  - text mining
  - visualization (bar, scatter)
  - advanced modeling



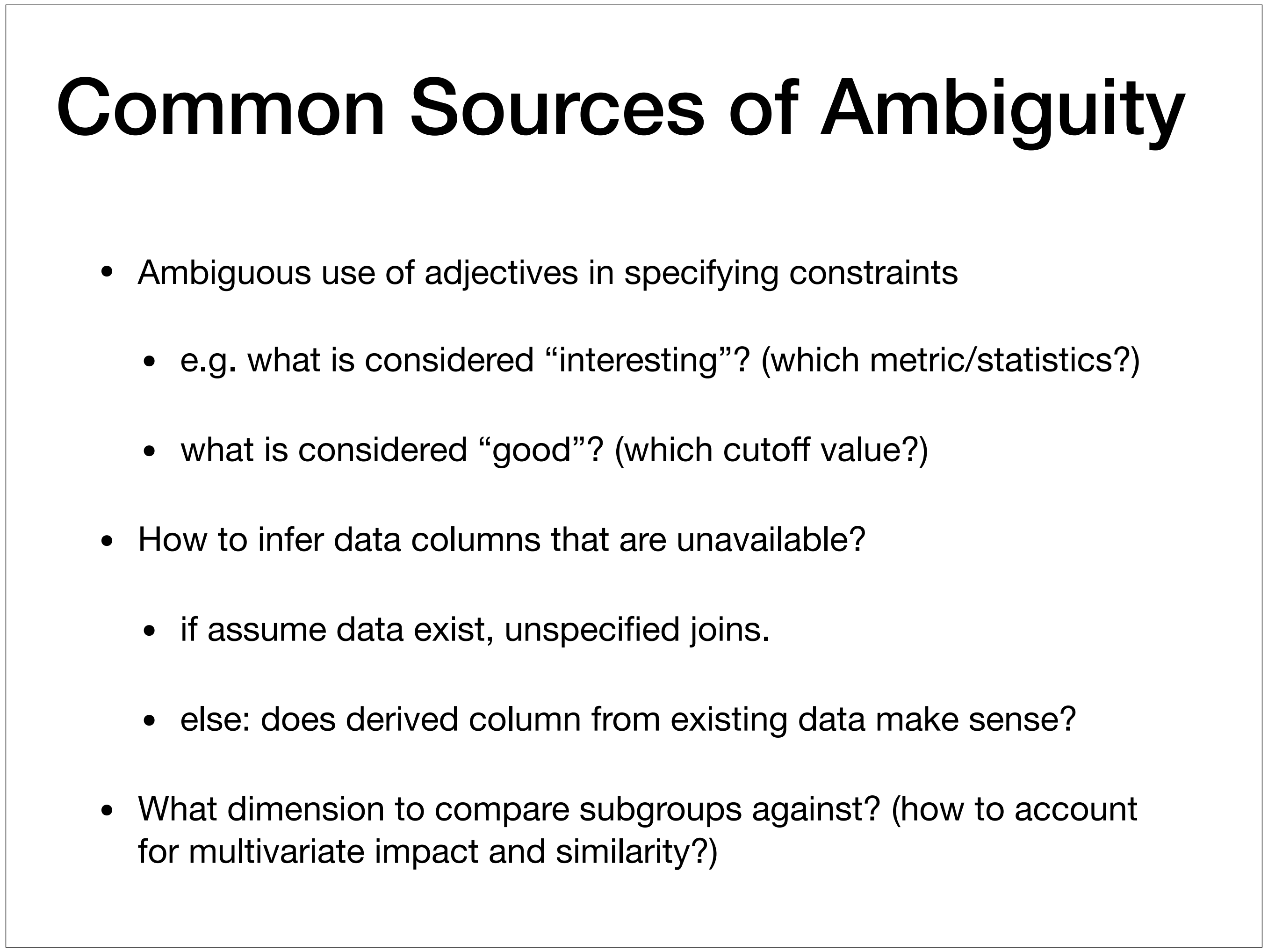
So far, I've collected 22 example scenarios that participants have come up with. These scenarios are fairly realistic and diverse in terms of the roles of the analysts, the analysis objective, and types of queries and tasks that they were interested in. My goal is to perform a thematic analysis on the descriptions of these analysis goals and task to extract design specifications for PQL. Here's some of the task summary showing that most tasks involve multiple steps, involving many of the actions listed here, from basic SQL operations to visualizations.



Common Goals

- Compare […] across different groups (6)
- Find “best” […] while satisfying multiple constraints and preferences (5)
- Understand relationships between multiple variables (2)
- Explore […]. Show me something interesting (2)
- Others: find representative, outliers, recommendation, budget allocation, summarization (1)

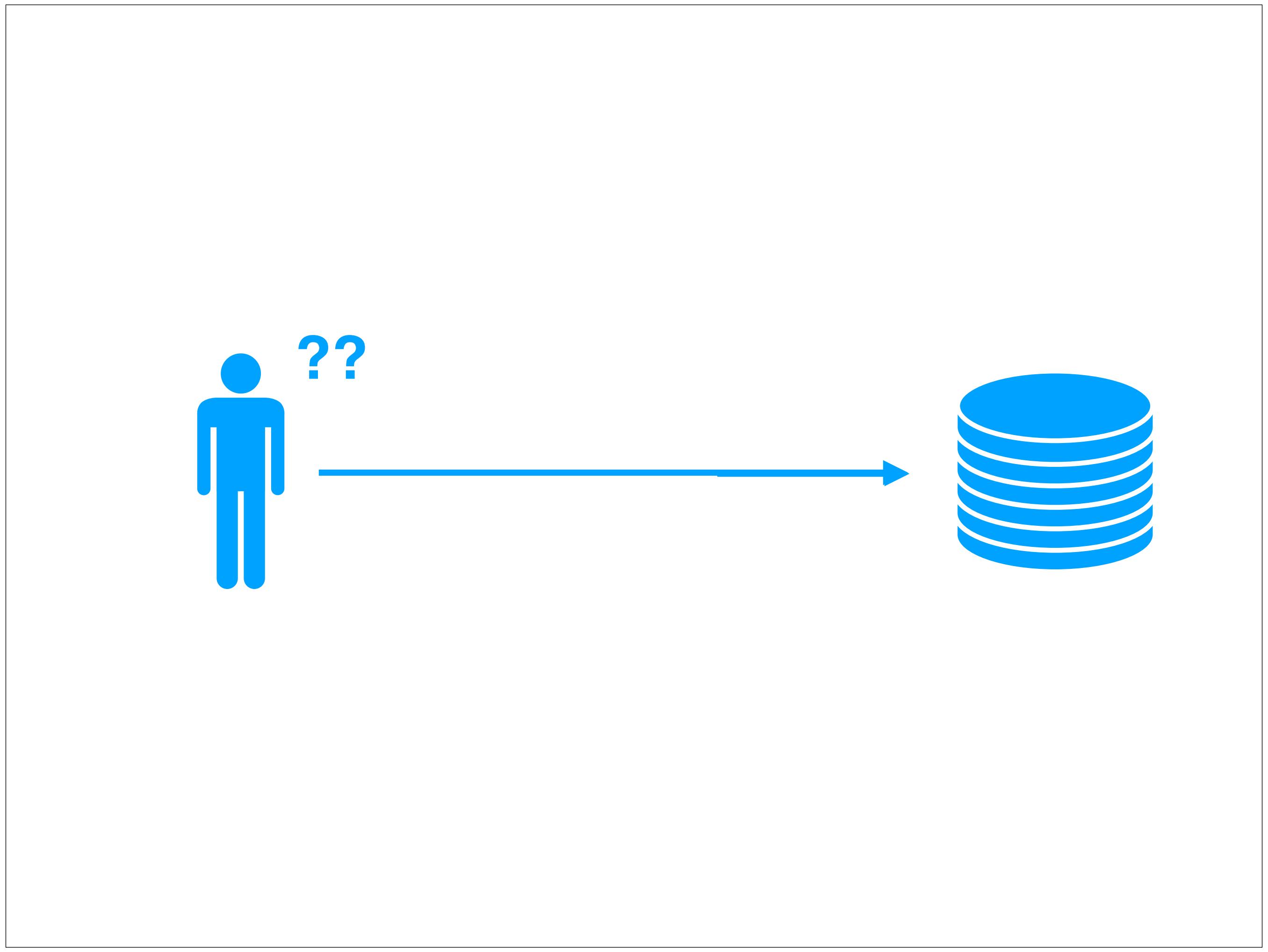
I’ve abstracted the scenario details away in the descriptions and found common themes across the analysis goals. In brackets are how common these goals are, listed in order from highest to lowest occurrence. Comparisons across groups and finding best based on multiple constraints and preferences were the most common tasks for this dataset. There were also some more exploratory tasks regarding relationship finding and generally finding something interesting.



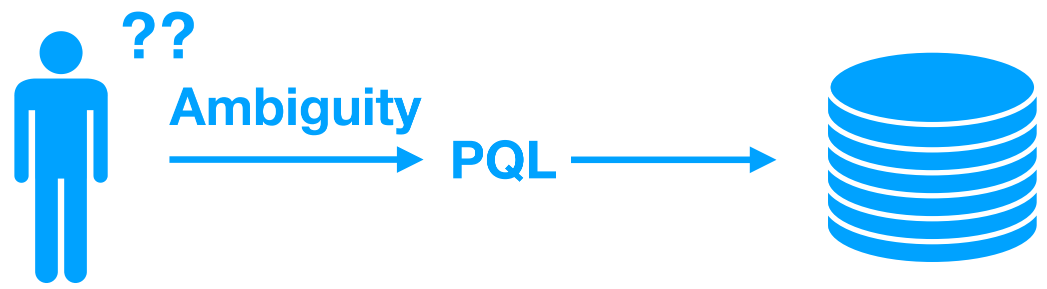
Common Sources of Ambiguity

- Ambiguous use of adjectives in specifying constraints
  - e.g. what is considered “interesting”? (which metric/statistics?)
  - what is considered “good”? (which cutoff value?)
- How to infer data columns that are unavailable?
  - if assume data exist, unspecified joins.
  - else: does derived column from existing data make sense?
- What dimension to compare subgroups against? (how to account for multivariate impact and similarity?)

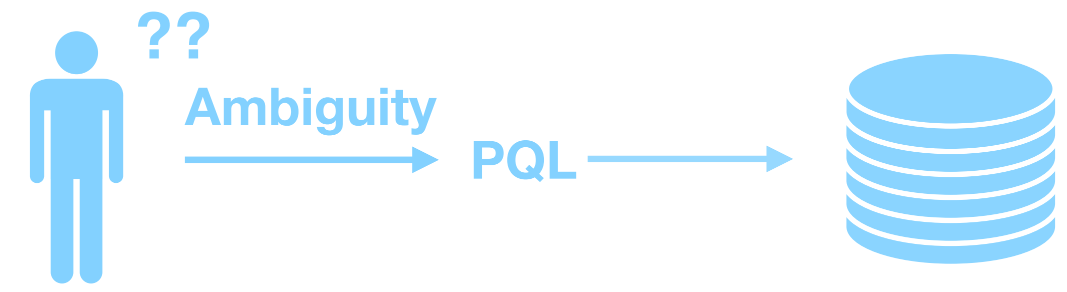
During task specification, there was many levels of ambiguity. One of the most common source of ambiguity is the use of adjectives in specifying constraints. The ambiguity can come in both sides of the inequality constraints , what metrics should one use for considering something as “interesting” and even if we find that metric (which can sometimes simply be a column), what is the cutoff value you should set for defining something as “good enough”.
Another common point of ambiguity comes in when some information of interest required for the analysis is not available in the data table. In this case, the participants either works with the assumption that this information exist somewhere and can be used alongside the existing information. We found it interesting that none of the participants specified join as a step in these analysis, which tells us that join operators is something people don’t really think about or assume should be done automatically. In the other case, if they don’t assume the data exist in another table, they often use some heuristic to derive this information from existing columns. These heuristics often introduce assumptions and the results needs to be checked for validity. Finally, other sources of ambiguous comes in when considering influences of multiple variables across different subgroups.



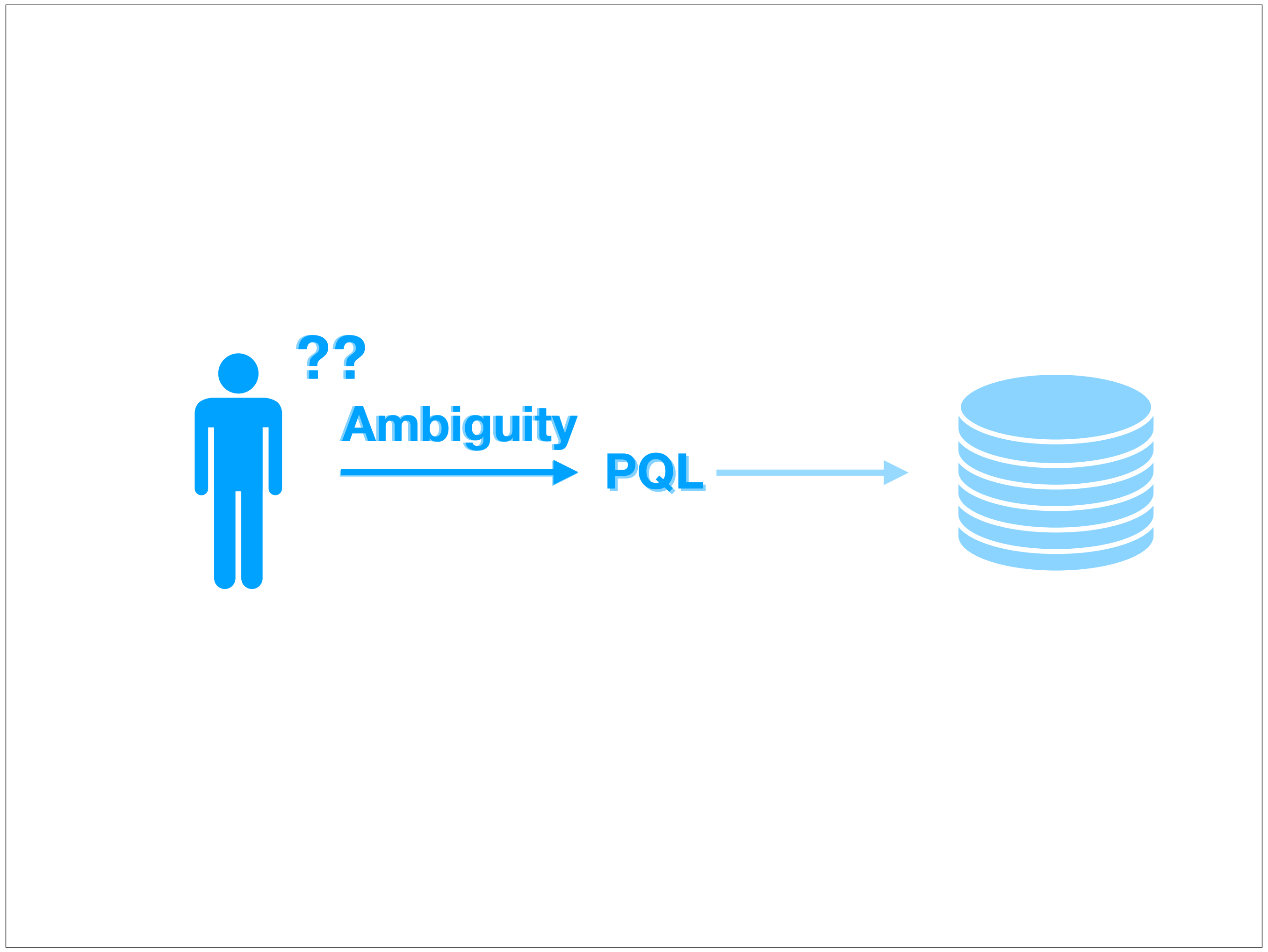
To conclude, I started from the problem of how to query a database. [*] My goal is to build a querying language that tolerates ambiguity to help people perform quick queries and getting to the point where they can interpret the returned result, without being boggled down by the semantics of **how to** query. [*,*] In the course project, I've focussed on the portion of studying how people specify ambiguous queries through a formative study. [*]As future work, I'm looking into designing a complete set of functions, operators and rules for the PQL language that fits these requirements.



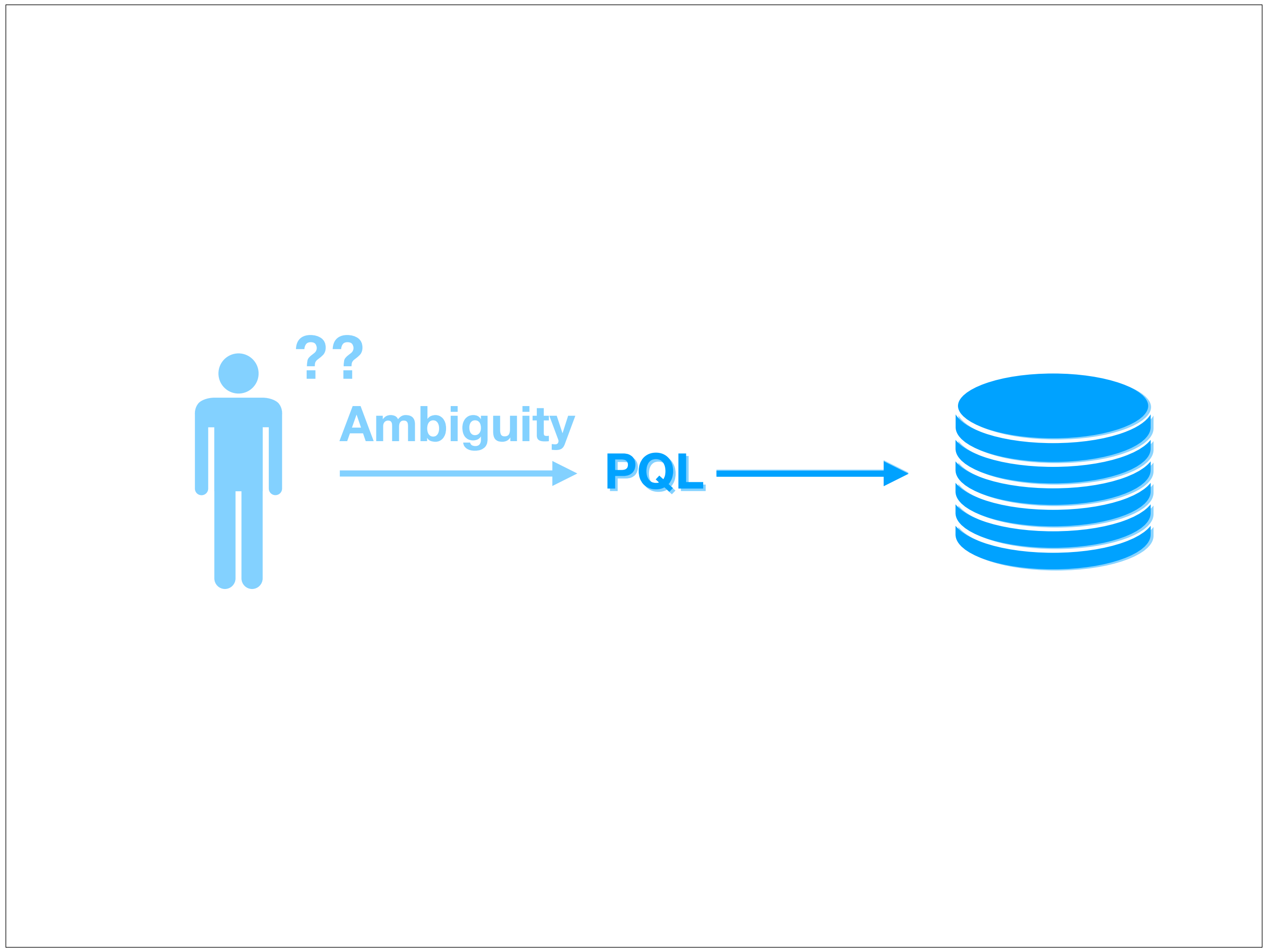
To conclude, I started from the problem of how to query a database. [*] My goal is to build a querying language that tolerates ambiguity to help people perform quick queries and getting to the point where they can interpret the returned result, without being boggled down by the semantics of **how to** query. [*,*] In the course project, I've focussed on the portion of studying how people specify ambiguous queries through a formative study. [*]As future work, I'm looking into designing a complete set of functions, operators and rules for the PQL language that fits these requirements.



To conclude, I started from the problem of how to query a database. [*] My goal is to build a querying language that tolerates ambiguity to help people perform quick queries and getting to the point where they can interpret the returned result, without being boggled down by the semantics of **how to** query. [*,*] In the course project, I've focussed on the portion of studying how people specify ambiguous queries through a formative study. [*]As future work, I'm looking into designing a complete set of functions, operators and rules for the PQL language that fits these requirements.



To conclude, I started from the problem of how to query a database. [*] My goal is to build a querying language that tolerates ambiguity to help people perform quick queries and getting to the point where they can interpret the returned result, without being boggled down by the semantics of **how to** query. [*,*] In the course project, I’ve focussed on the portion of studying how people specify ambiguous queries through a formative study. [*]As future work, I’m looking into designing a complete set of functions, operators and rules for the PQL language that fits these requirements.



To conclude, I started from the problem of how to query a database. [*] My goal is to build a querying language that tolerates ambiguity to help people perform quick queries and getting to the point where they can interpret the returned result, without being boggled down by the semantics of **how to** query. [*,*] In the course project, I've focussed on the portion of studying how people specify ambiguous queries through a formative study. [*]As future work, I'm looking into designing a complete set of functions, operators and rules for the PQL language that fits these requirements.