Applications for a standard DBSM

End User 1 → Application process 1

End User 2 → Application process 2

End User 2 → Application process 3

DBSM

Database
Interactive Data Exploration
Applications

End User 1 -> IDE Application 1 -> Big Data / standard DBSM

End User 2 -> IDE Application 2

End User 2 -> IDE Application 3
Using IDE

As the datasets are so complex, users would need an “expert assistant” to help them.

- This requires an automated process

DBNav
# IDE with DBNav: a quick analogy

Driving:

<table>
<thead>
<tr>
<th>Without navigation help</th>
<th>With navigation help</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Paper map</td>
<td>• Guides driver to destination with turn-by-turn directions</td>
</tr>
<tr>
<td>• Stop to ask for information</td>
<td>• Shows current location in real time, along with points of interest</td>
</tr>
<tr>
<td></td>
<td>• Gives arrival estimate time, pointing possible delay interferences on the way</td>
</tr>
</tbody>
</table>
Functionalities addressed by IDE

✓ Interactive performance: online, interruptible and progressive queries that allows more effective data exploration

✓ Navigation help: provide recommendations that guide users towards more interesting parts of data

✓ Visualization: effective presentations help identification of interesting features
✓ **Personalization and customization**: models should fit users' interests and goals.

And also:

✓ **“Query session”**: a sequence of related queries, each one serving as a jumping-off point to the next.

✓ **User and application profiles**: rich user models and interfaces that allow users to provide feedback to the system or models.
We will be discussing the two main DBNavs in the next few slides:

- Smart Drill-Down
- Query Steering
Smart Drill-Down

• Goal: To discover and summarize interesting group of tuples using smart drill down operator. (keeping in mind the fundamental concepts of drill down).

• Group of tuples can be described as a rule.
  - e.g (a, b, *, 1000)

• A simple explanation to the above mention rule will be thousand tuples with the a in first col, b in second col, anything in third, total count 1000 for tuples.
Problems with traditional drill down

- When the distinct values in a column are large, drill down would end up showing too many results (i.e., aggregates).
- Drill down usually applied on one column, multiple simultaneous drill down on multiple columns would again end up in too many results.
- Lets us to zoom into more interesting parts of the table or database without having to examine as much data as traditional drill down.
- Example: Consider cols: dept, store, prod, reg, sales,
- Goal is to see the stores where sales were higher than a certain threshold.
- Initial result is (*, *, * 6000, 0). (* is wild card matching any value)
- Count attribute can be replaced by sum aggregate as we want top sales.
First drill-down (smart v traditional)

- Now if we drill down on first * that is a store. Operator displays the tuples of the form \((X, *, *, C, 1)\)
- \(X\) is store, \(C\) is the number of tuples or aggregate sales score for \(X\).
- Instead when small drill down is used, rule \((*, *, *, 6000)\) is expanded into 3 tuples that displays interesting drill down.
- The number 3 is user specified parameter.
1st Smart-Drill

**TABLE I: Initial Summary**

<table>
<thead>
<tr>
<th>Store</th>
<th>Product</th>
<th>Region</th>
<th>Count</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>*</td>
<td>*</td>
<td>6000</td>
<td>0</td>
</tr>
</tbody>
</table>

**TABLE II: Result After First Smart Drill Down**

<table>
<thead>
<tr>
<th>Store</th>
<th>Product</th>
<th>Region</th>
<th>Count</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>*</td>
<td>*</td>
<td>6000</td>
<td>0</td>
</tr>
<tr>
<td>▶ Target</td>
<td>bicycles</td>
<td>*</td>
<td>200</td>
<td>2</td>
</tr>
<tr>
<td>▶ *</td>
<td>comforters</td>
<td>MA-3</td>
<td>600</td>
<td>2</td>
</tr>
<tr>
<td>▶ Walmart</td>
<td>*</td>
<td>*</td>
<td>1000</td>
<td>1</td>
</tr>
</tbody>
</table>
1\textsuperscript{st} Smart-Drill

- Lets examine tuple (target, bicycles, *, 200, 2)
- It says, there are 200 tuples with first column value target and 2\textsuperscript{nd} column values as bicycle.
- Fact tells us that target is selling lot of bicycles.

- Lets dig a bit deeper to see in which states Walmart has better sales.
**TABLE III: Result After Second Smart Drill Down**

<table>
<thead>
<tr>
<th>Store</th>
<th>Product</th>
<th>Region</th>
<th>Count</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>*</td>
<td>*</td>
<td>6000</td>
<td>0</td>
</tr>
<tr>
<td>▶ Target</td>
<td>bicycles</td>
<td>*</td>
<td>200</td>
<td>2</td>
</tr>
<tr>
<td>▶ *</td>
<td>comforters</td>
<td>MA-3</td>
<td>600</td>
<td>2</td>
</tr>
<tr>
<td>▶ Walmart</td>
<td>*</td>
<td>*</td>
<td>1000</td>
<td>1</td>
</tr>
<tr>
<td>▶ ▶ Walmart</td>
<td>cookies</td>
<td>*</td>
<td>200</td>
<td>2</td>
</tr>
<tr>
<td>▶ ▶ Walmart</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>▶ ▶ ▶ Walmart</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>▶ ▶ ▶ Walmart</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Factors that make rules Interesting

- High Count/Sales (or any aggregate for that matter)
- Diversity

For example (target, bicycles, * 200, 2) would be preferred to (Walmart, bicycles, *, 200, 2) as we already have the more generic rule related to the later.

- Smart drill down explores several attributes to open up and selects that are interesting.
Advantages

- The rule (target, bicycles, *, 200, 2) is obtained after single drill down. In traditional drill down we would have first drill down on store → then to product, looked through all displayed rules to find interesting rule (target, bicycles, *, 200, 2)
Properties related to rules

- **Coverage**
  Rule r covers tuple t if all non* values matches value in tuple.

- **Rule Lists**
  Ordered list of rules, returned by system in response to a drill down.

- **Scoring**
  Two portions
  - How much the rule r covers the tuples t in scheme D.
  - How good rule r is (independent of how many tuples it covers)
Scoring

- Count(r) → as the total number of tuples $t \in T$ that are covered by r.
- MCount(r, R) (which stands for ‘Marginal Count’) as the number of tuples covered by r but not by any rule before r in the rule-list R.
- A high value of MCount indicates that the rule not only covers a lot of tuples, but also covers parts of the table not covered by previous rules.
- We want to pick rules with a high value of MCount.
• the second portion: we let $W$ denote a function that assigns a non-negative \textit{weight} to a rule based on how good the rule is, with higher weights assigned to better rules.

• Scoring function:

$$\text{Score}(R) = \text{sum} \left( \text{MCount}(r,R) \times W(r) \right)$$

$r \in R$ coverage of \( r \) in \( D \), weight of \( r \)
Next we will discuss Query Steering
Query Steering

- Process of assisting a user to navigate through a complex data space.
  - Assistance to the user (recommendations)
  - Faster process of the queries.

- Important: Query session (i.e. sequence of queries) generated by:
  - by the user
  - by the system (learning)
Query Steering Modes

3 Steering modes (in increasing order of the system involvement):

1. Manual Steering
2. Power Steering
3. Auto Steering
**Manual steering**: The user tweaks the parameter values.

- The user manually specifies the queries in the session one by one e.g.
  - Drill down
  - Move
  - Narrow
  - Relate
  - ...

Query Steering Modes
**Power Steering**

The user can specify:

- An arbitrarily long prioritized query sequence at once.

- Steering goals
  - e.g. ask for “the largest region in the sky where the average blue surface brightness value is higher than 24”
Auto Steering

The system:

- builds a user or application profile
- Investigates the data space on behalf of the user
- Automatically recommends queries
Query Steering Operations

e.g.
- Drill down
- Move
- Narrow

TRUE: Other systems support similar navigational activities (e.g. OLAP)

Distinction: Our focus is on the optimization of queries based on the user behavior and interaction profiles.
Query steering: Navigational Profile

Example:

- **Start**
- **Drill down**
- **Roll up**
- **Move**

- **Probablity models**
  - Probability models
  - Sequential association rules
  - 2 → 3, 5 (.8)
  - 2 → 6, 5 (.2)
  - 7 → 1 (.2)
  - ...

- **Transition Probabilities**
  - 0.1
  - 0.2
  - 0.7
Objective:
Optimization of Query sessions:
  Fast Processing + User Assistance

➔ THE BIG QUESTION:
   How?
Profile-driven prefetching and caching

- Prefetch more data than needed in Qi in anticipation for Qi+1

Prefetching: transfer (data) from main memory to temporary storage in readiness for later use.
Query checkpointing to facilitate reusable progress

- Take advantage of the overlap of queries within the same session

1. Break down queries into many checkpoint queries.
2. If similar checkpoint queries are found, save the data for later reuse.
Query steering: Optimizations

Query checkpointing to facilitate reusable progress

- Take advantage of the overlap of queries within the same session
- Example:

C2 and C’3 are similar checkpoints
→ cache the result of C2 to reuse in C’3
Query Steering: Optimizations

Query sequence optimizations (power steering)

- Take advantage of correlated queries
- Reuse-oriented techniques.
Efficient query learning (auto steering)

- the user simply characterizes data samples and the system automatically formulates « classification » queries.

  → Learning
Conclusion

Two main DBNavs are presented:

- **Query Steering**: Process of assisting the user navigate complex data.
- **Manual Steering (No Machine Assist)**
- **Power Steering (Partial Assist)**
- **Auto Steering (Queries are automatically generated)**

- **Smart Drill-Down**: Allows an analyst to discover new patterns (rules) in existing data sources.
- **Faster to use than traditional drill down**
- **Do not need to traverse entire table**
Thus we have illustrated the viability of using interactive data exploration techniques like **smart drill down** and **query steering** to enable a user to recognize new patterns in complex data using visual cues.
Thank You
Any Questions?