Tiresias: Enabling Predictive Autonomous Storage and Indexing

Michael Abebe  mtabebe@uwaterloo.ca
Horatiu Lazu
Khuzaima Daudjee

September 2022
tiny.cc/tiresias
DBMS storage & indexing choices have trade-offs based on workload
## Index Choice

<table>
<thead>
<tr>
<th>Order</th>
<th>Book</th>
<th>User</th>
<th>Qnt</th>
<th>$</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>HP</td>
<td>Alice</td>
<td>7</td>
<td>$15</td>
</tr>
<tr>
<td>101</td>
<td>Drac</td>
<td>Ted</td>
<td>90</td>
<td>$3</td>
</tr>
<tr>
<td>102</td>
<td>SQL</td>
<td>Geoff</td>
<td>1</td>
<td>$100</td>
</tr>
<tr>
<td>103</td>
<td>HP</td>
<td>Jean</td>
<td>3</td>
<td>$15</td>
</tr>
</tbody>
</table>

**SELECT * WHERE QNT > 15**

Should we index?
Index Choice

SELECT * WHERE QNT > 15

Should we index? Yes

Lower is better

Query Lat. (ms)

No Index | Index

0 | 30
30 | 90

UNIVERSITY OF WATERLOO
# Index Choice

<table>
<thead>
<tr>
<th>Order</th>
<th>Book</th>
<th>User</th>
<th>Qnt</th>
<th>$</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>HP</td>
<td>Alice</td>
<td>7</td>
<td>$15</td>
</tr>
<tr>
<td>101</td>
<td>Drac</td>
<td>Ted</td>
<td>90</td>
<td>$3</td>
</tr>
<tr>
<td>102</td>
<td>SQL</td>
<td>Geoff</td>
<td>1</td>
<td>$100</td>
</tr>
<tr>
<td>103</td>
<td>HP</td>
<td>Jean</td>
<td>3</td>
<td>$15</td>
</tr>
<tr>
<td>104</td>
<td>LoTR</td>
<td>John</td>
<td>2</td>
<td>$21</td>
</tr>
</tbody>
</table>

**INSERT** (104, LoTR, John, 2, $21)

Should we index?
**Index Choice**

Higher is better

**INSERT** (104, LoTR, John, 2, $21)

Should we index?  
No
## Storage Choice

<table>
<thead>
<tr>
<th>Order</th>
<th>Book</th>
<th>User</th>
<th>Qnt</th>
<th>$</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>HP</td>
<td>Alice</td>
<td>7</td>
<td>$15</td>
</tr>
<tr>
<td>101</td>
<td>Drac</td>
<td>Ted</td>
<td>90</td>
<td>$3</td>
</tr>
<tr>
<td>102</td>
<td>SQL</td>
<td>Geoff</td>
<td>1</td>
<td>$100</td>
</tr>
<tr>
<td>103</td>
<td>HP</td>
<td>Jean</td>
<td>3</td>
<td>$15</td>
</tr>
<tr>
<td>104</td>
<td>LoTR</td>
<td>John</td>
<td>2</td>
<td>$21</td>
</tr>
</tbody>
</table>

**INSERT** (104, LoTR, John, 2, $21)
Storage Choice

Higher is better

INSERT (104, LoTR, John, 2, $21)

How should we store?
Row
### Storage Choice

<table>
<thead>
<tr>
<th>Order</th>
<th>Book</th>
<th>User</th>
<th>Qnt</th>
<th>$</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>HP</td>
<td>Alice</td>
<td>7</td>
<td>$15</td>
</tr>
<tr>
<td>101</td>
<td>Drac</td>
<td>Ted</td>
<td>90</td>
<td>$3</td>
</tr>
<tr>
<td>102</td>
<td>SQL</td>
<td>Geoff</td>
<td>1</td>
<td>$100</td>
</tr>
<tr>
<td>103</td>
<td>HP</td>
<td>Jean</td>
<td>3</td>
<td>$15</td>
</tr>
<tr>
<td>104</td>
<td>LoTR</td>
<td>John</td>
<td>2</td>
<td>$21</td>
</tr>
</tbody>
</table>

**SELECT * WHERE QNT > 15**
Storage Choice

**SELECT** * WHERE
QNT > 15

How should we store?
- Column

![Bar chart showing OLAP Lat. (ms) for Column and Row storage types. Lower is better.](chart.png)
DBMS storage & indexing choices have trade-offs based on workload so choices should be adaptive!
Workloads Change
Tiresias

**Predict** upcoming accesses and latency under different storage and indexing choices

**Tiresias**
- **Model** workload access history
- **Model** access latencies

**DBMS**
- **Store and index data**
- **Change** data storage and indexing

Report query latency and access history
Tiresias

**Predict** upcoming accesses and latency under different storage and indexing choices

- **Tiresias**
  - Model workload *access history*
  - Model *access latencies*

- **DBMS**
  - PostgreSQL: automatic indexing
  - OLAP DBMS: predictive cracking
  - Proteus: adaptive storage for HTAP

Report query latency and access history
Tiresias

Predict upcoming accesses and latency under different storage and indexing choices

1. Model workload access history
2. Model access latencies
3. Change data storage and indexing

Report query latency and access history
Access Arrival Patterns
Access Arrival Patterns

Daily

Weekly

Requests per Minute

Wikipedia Trace
Learning Access Arrival Patterns

Requests per Minute

Date

Learning Access Arrival Patterns

\[ \text{SPAR} = \text{Periodicity} + \text{Trend} \]

**SPAR**

**Periodicity**
Avg. over user defined intervals

**Trend**
Avg. over interval

**Hybrid-Ensemble**

**Recurrent Neural Network**

**Linear Regression**
Access Arrival Patterns - Results

Hybrid-Ensemble

Requests per Minute

0K  50K  100K  150K  200K


Date
Access Arrival Patterns - Results

SPAR

Requests per Minute

Date

Access Arrival Patterns - Accuracy

RMSE (%)

Lower is better

- SPAR
- Hybrid Ensemble
- No Prediction (Avg)
Access Arrival Patterns - Results

SPAR (Mis-Configured)

Incorrect definition of period decreases accuracy!
Tiresias

Predict upcoming accesses and latency under different storage and indexing choices

1. Tiresias
   - Model workload access history
   - Model access latencies

2. DBMS
   - Store and index data
   - Change data storage and indexing

3. Report query latency and access history
Predicting Latency

Transactions are composed of physical operators

```sql
SELECT book, SUM(qnt) GROUP BY book
```

**Row layout**
- Row scan
- Hash aggregation: `book, sum(qnt)`

**Logical Plan**
- Scan & Project: `book, qnt`
- Aggregate: `book, sum(qnt)`
- Sorted col aggregation: `book, sum(qnt)`

**Sorted column layout**
- Sequential col scan
Predicting Latency

Transactions are composed of physical operators

Predict physical operator latency
Per layout with workload stats as parameters

Cardinality
Data Width \rightarrow \text{Predicted Latency}

Est Selectivity \rightarrow \text{Seq col scan}
Predicting Latency

Transactions are composed of physical operators

Predict physical operator latency

Per layout with workload stats as parameters

Cardinality

Data Width

Est Selectivity

Predicted Latency

Row scan
Predicting Latency

Transactions are composed of physical operators

Predict physical operator latency

Per layout with workload stats as parameters

Linear Regressor

Neural Network

Non-Linear Regressor
Predicting Latency Accuracy

Lower is better

Linear Regressor
Neural Network
Non-Linear Regressor
Predicting Latency - Training

![Bar chart showing comparison between Non-Linear Regressor, Neural Network, and Linear Regressor. Lower is better.](chart)

- Non-Linear Regressor
- Neural Network
- Linear Regressor

Lower is better
Predicting Latency - Inference

Lower is better
Tiresias

Predict upcoming accesses and latency under different storage and indexing choices

1. Tiresias
   - Model workload access history
   - Model access latencies

2. Report query latency and access history

3. DBMS
   - Store and index data
   - Change data storage and indexing
Storage and Index Changes

Expected Benefit of Change

Predicted access latency under: current & proposed storage/indexing choice

Weighted by likelihood of access

Cost incurred to perform change
Tiresias End-to-End Evaluation

DBMS

**PostgreSQL:** automatic indexing

**OLAP DBMS:** predictive cracking

**Proteus:** adaptive storage for HTAP

Adapt with Tiresias

Static choices without Tiresias

Shifting Workloads

Skew (hotspots)

Mix (OLTP or OLAP heavy)
End-to-End PostgreSQL Indexes

Higher is better

No Index

Index

Lower is better

No Index

Index
End-to-End PostgreSQL Indexes

Higher is better

No Index
Adaptive - Tiresias Index

Lower is better

No Index
Adaptive - Tiresias Index
End-to-End PostgreSQL Indexes

Lower is better
No Index
Adaptive - Tiresias Index
15% & 70% time reduction

Workload Comp. Time (sec.)

Overall  OLTP Heavy Workload  OLAP Heavy
End-to-End Cracking

Lower is better

38x latency reduction

Cracking (No Tiresias)
Predictive Cracking: Tiresias
End-to-End Proteus

Lower is better

28% improvement
End-to-End Proteus

Higher is better

OLTP Tput. (TPS)

0K 4K 8K 12K 16K

Column  Row  Proteus + Tiresias  Proteus -Tiresias
End-to-End Proteus

![Bar chart showing OLAP Latency (ms) for Column, Row, Proteus + Tiresias, and Proteus - Tiresias. Lower is better.](chart)
Tiresias Takeaways tiny.cc/tiresias

Automatic adaptation of data storage and indexing

Generalizable API used in different DBMSs

Predict data accesses based on workload patterns

Learn access costs under different storage choices