Microsoft Azure Application Insights

SQL to Analytics language cheat sheet

Download this document at https://aka.ms/sql-analytics

<table>
<tr><th></th><th>SQL Query</th><th>Analytics Query</th></tr>
<tr><td rowspan="4">Select data from table</td><td>SELECT * FROM dependencies</td><td>dependencies</td></tr>
<tr><td rowspan="2">SELECT name, resultCode FROM dependencies</td><td>dependencies | project name, resultCode</td></tr>
<tr><td>dependencies | project-away name</td></tr>
<tr><td>SELECT TOP 100 * FROM dependencies</td><td>dependencies | take 100</td></tr>
<tr><td>Null evaluation</td><td>SELECT * FROM dependencies
WHERE resultCode IS NOT NULL</td><td>dependencies | where isnotnull(resultCode)</td></tr>
<tr><td rowspan="2">Comparison operators (date)</td><td>SELECT * FROM dependencies
WHERE timestamp > getdate()-1</td><td>dependencies
| where timestamp > ago(1d)</td></tr>
<tr><td>SELECT * FROM dependencies
WHERE timestamp
      BETWEEN '2016-10-01' AND '2016-11-01'</td><td>dependencies
| where timestamp > datetime(2016-10-01)
        and timestamp <= datetime(2016-11-01)</td></tr>
<tr><td rowspan="3">Comparison Operators (string)</td><td>SELECT * FROM dependencies
WHERE type = "Azure blob"</td><td>dependencies
| where type == "Azure blob"</td></tr>
<tr><td>--substring
SELECT * FROM dependencies
WHERE type like "%blob%"</td><td>//substring
dependencies
| where type contains "blob"</td></tr>
<tr><td>--wildcard
SELECT * FROM dependencies
WHERE type like "Azure%"</td><td>dependencies
| where type startswith "Azure"

dependencies
| where type matches regex "^Azure.*"</td></tr>
<tr><td>Comparison (boolean)</td><td>SELECT * FROM dependencies
WHERE !(success)</td><td>dependencies
| where success == "False"</td></tr>
<tr><td>Distinct</td><td>SELECT DISTINCT name, type
FROM dependencies</td><td>dependencies
| summarize by name, type</td></tr>
<tr><td>Grouping, Aggregation</td><td>SELECT name, AVG(duration)
FROM dependencies
GROUP BY name</td><td>dependencies
| summarize avg(duration) by name</td></tr>
<tr><td>Column aliases, Extending</td><td>SELECT operation_Name as Name,
       AVG(duration) as AvgD
FROM dependencies
GROUP BY name</td><td>dependencies
| summarize AvgD=avg(duration) by operation_Name
| project Name=operation_Name, AvgD</td></tr>
<tr><td>Ordering</td><td>SELECT name, timestamp
FROM dependencies
ORDER BY timestamp asc</td><td>dependencies
| project name, timestamp
| order by timestamp asc nulls last</td></tr>
<tr><td>Top n by measure</td><td>SELECT TOP 100 name,
       COUNT(*) as Count
FROM dependencies
GROUP BY name
ORDER BY Count desc</td><td>dependencies
| summarize Count=count() by name
| top 100 by Count desc</td></tr>
<tr><td rowspan="2">Union</td><td>SELECT * FROM dependencies
UNION
SELECT * FROM exceptions</td><td>union dependencies, exceptions</td></tr>
<tr><td>SELECT * FROM dependencies WHERE timestamp>..
UNION
SELECT * FROM exceptions WHERE timestamp>..</td><td>dependencies | where timestamp > ago(1d)
| union
  (exceptions | where timestamp > ago(1d))</td></tr>
<tr><td>Join</td><td>SELECT * FROM dependencies
LEFT OUTER JOIN exception
ON dependencies.operation_Id =
   exceptions.operation_Id</td><td>dependencies
| join kind=leftouter
   (exceptions)
  on $left.operation_Id == $right.operation_Id</td></tr>
</table>

These are just subset of the operators available. Please refer to https://aka.ms/AIAnalyticsReference for a complete reference.

Try Analytics yourself by instrumenting with Azure Application Insights, or in the Analytics demo environment: https://aka.ms/AIAnalyticsDemo!

Microsoft

https://www.azure.com

AIAnalyticsOutreach@microsoft.com

@Azure

Application Insights Analytics – useful operators

Category	Relevant Analytics functions
Selection and Column aliases	project, project-away, extend
Temporary tables and constants	let scalar_alias_name = ...; let table_alias_name = (){ ... | ... | ... };
Comparison and String Operators	startswith, !startswith has*, !has contains, !contains, containscs hasprefix, !hasprefix, hassuffix, !hassuffix in, !in matches regex ==, =~, !=, !~ *_has_ is more performant than _contains_

Common string functions	strcat(), replace()
	tolower()*, toupper()*
	substring(), strlen()
	
	*for a more performant solution than converting case when comparing strings use:
	"aBc" =~ "abc"

Common math functions	sqrt(), abs()
	exp(), exp2(), exp10(), log(), log2(), log10()
	pow()
	gamma(), gammaln()

Parsing text	extract(), extractjson(), parse*, split() *parse is more performant
Limiting output	take, limit, top, sample hash

Date functions	now(), ago()
	datetime(), datepart(), timespan
	startofday(), startofweek(), startofmonth(), startofyear()
	endofday(), endofweek(), endofmonth(), endofyear()
	dayofweek(), dayofmonth(), dayofyear()
	getmonth(), getyear(),
	weekofyear(), monthofyear()

Grouping and aggregation
 	
by
top, count(), min(), max(), bin()
	summarize by
	max(), min(), count(), dcount(), avg(), sum(), stddev()
	countif(), dcountif()
	argmax(), argmin()
	percentiles(), percentile_array()
	
	top, top-nested

Joins and Unions	join kind=leftouter, inner, rightouter, fullouter, leftanti union
Sort, order	sort, order

Dynamic object (JSON and array) operators and functions	parsejson()
	makeset(), makelist()
	split(), arraylength()
	zip(), pack()

Logical operators	iff(condition, value_t, value_f)
	
	binary_and(), binary_or(), binary_not(), binary_xor()

Machine learning	evaluate autocluster, basket, diffpatterns, extractcolumns

More info about these and other functions and operators is available on our language reference: https://aka.ms/AIAnalyticsReference