PocketData Benchmark

[Week #2]

Naveen, Sankar, Saravanan, Sathish

Progress

- Classification based on business domain.
  - 173 applications.
  - 26 clusters
- Finding features (In Progress)
  - Read & Write percentage,
  - Bursts,
  - Complexity of queries etc.

26 Clusters

System Services Utility Music Pre-Installed Networking

e-Commerce Messaging & Calls Game

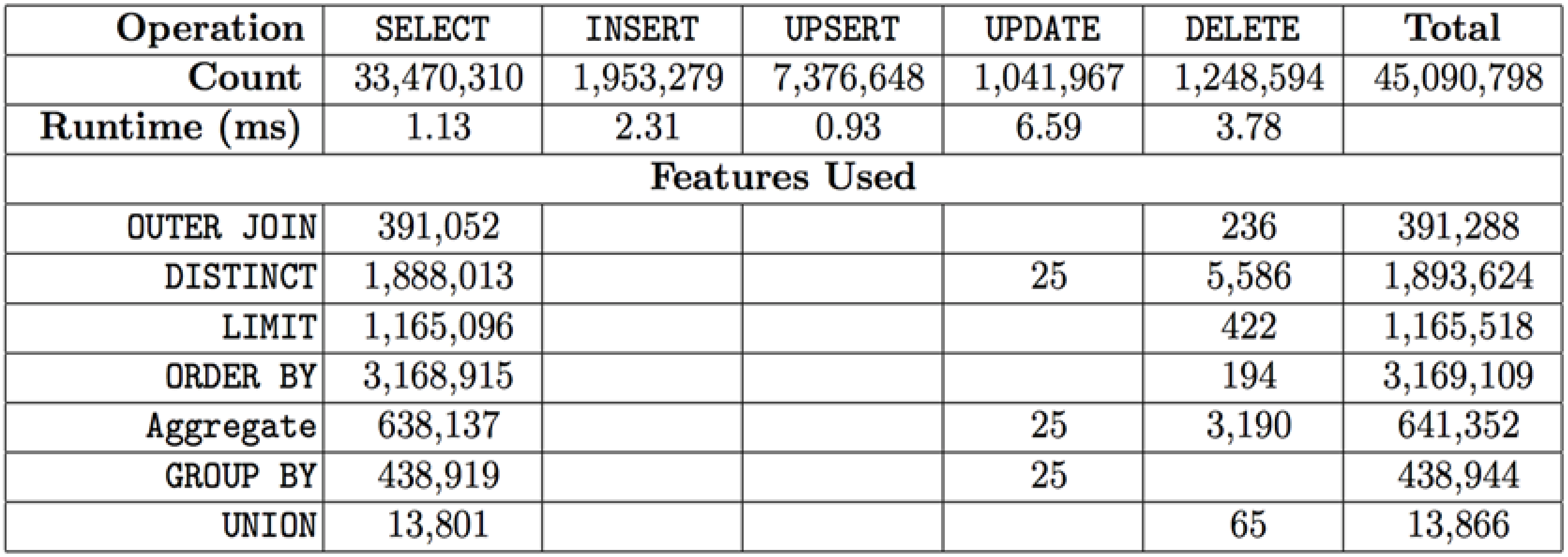
Cloud Video Services Browser Image Forum New/RSS Reader

Cloud Storage Email Search eReader Dating

Navigation Media Player | Finance Health | Programming | Productivity | Personalization | Antivirus | Miscellaneous

Observations from ‘Pocket Data: The Need for TPC-MOBILE’ paper

Types and numbers of SQL statements executed during the one-month trace

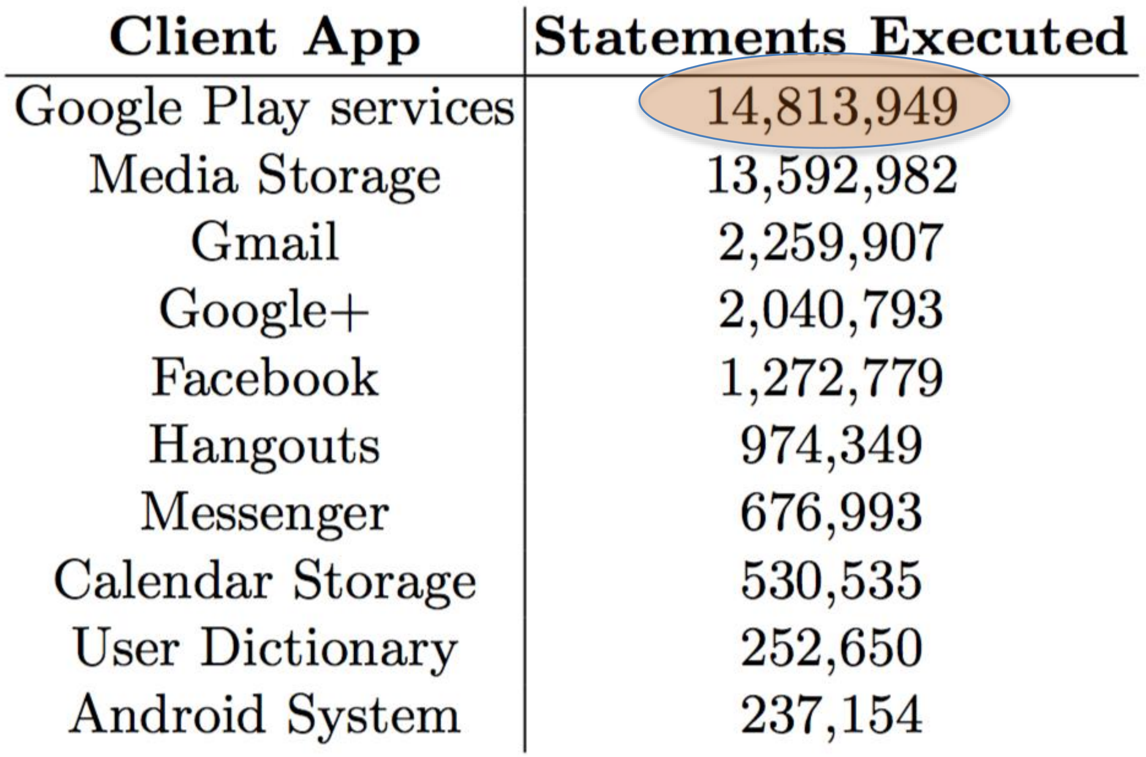


Operation	SELECT	INSERT	UPSERT	UPDATE	DELETE	Total
Count	33,470,310	1,953,279	7,376,648	1,041,967	1,248,594	45,090,798
Runtime (ms)	1.13	2.31	0.93	6.59	3.78	
Features Used						
OUTER JOIN	391,052				236	391,288
DISTINCT	1,888,013			25	5,586	1,893,624
LIMIT	1,165,096				422	1,165,518
ORDER BY	3,168,915				194	3,169,109
Aggregate	638,137			25	3,190	641,352
GROUP BY	438,919			25		438,944
UNION	13,801				65	13,866

- 74% Select | 71% of INSERT/UPDATE statements are UPSERTS
- ~10% Select has Order By | Unions seldom used
- Deletes are complex (Cache Invalidation: Invalidating the offline cache data as soon as it connects to internet?)

Reference: http://odin.cse.buffalo.edu/wp-content/uploads/2015/06/2015-TPCTC-SQLite-submitted.pdf

Observations from ‘Pocket Data: The Need for TPC-MOBILE’ paper



Client App	Statements Executed
Google Play services	14,813,949
Media Storage	13,592,982
Gmail	2,259,907
Google+	2,040,793
Facebook	1,272,779
Hangouts	974,349
Messenger	676,993
Calendar Storage	530,535
User Dictionary	252,650
Android System	237,154

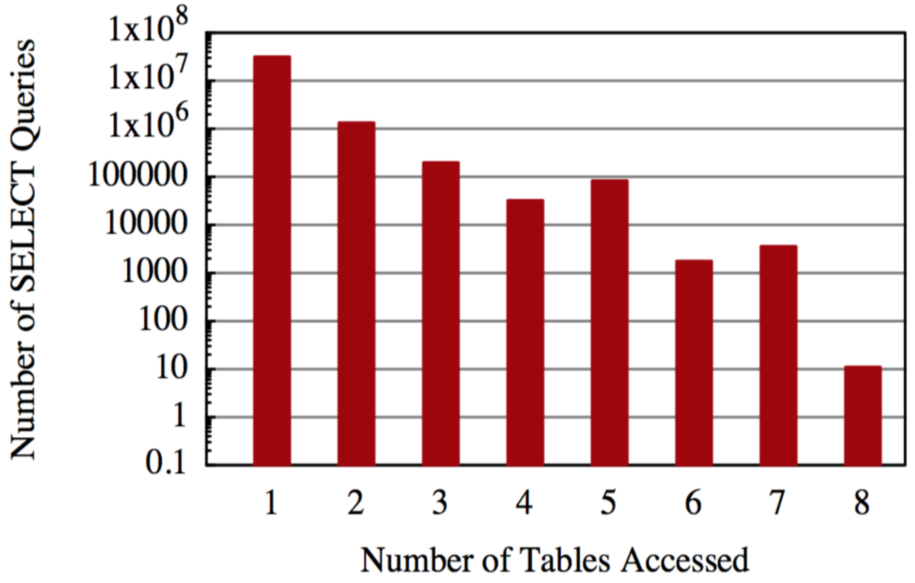
(a)

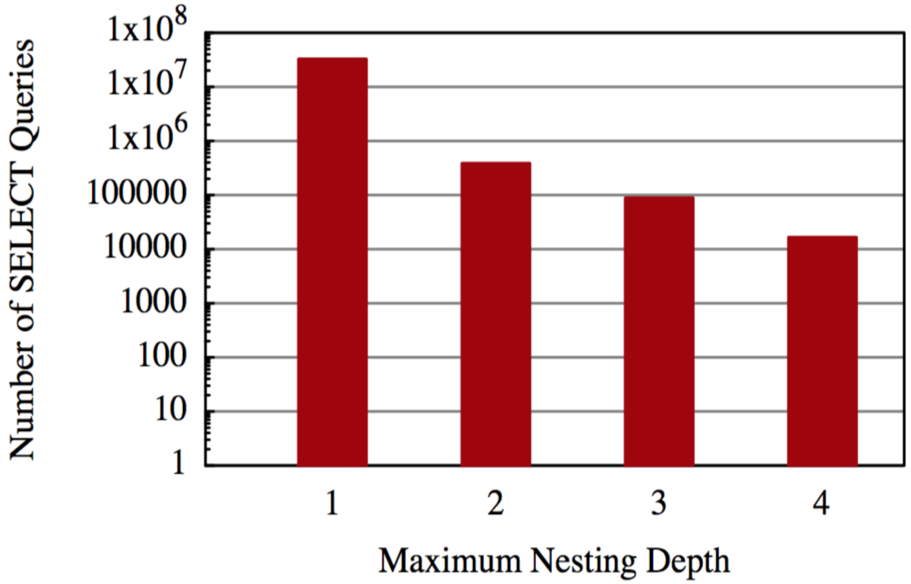
33% queries from a single service

63% queries summing up top two services

Reference: http://odin.cse.buffalo.edu/wp-content/uploads/2015/06/2015-TPCTC-SQLite-submitted.pdf

Observations from ‘Pocket Data: The Need for TPC-MOBILE’ paper

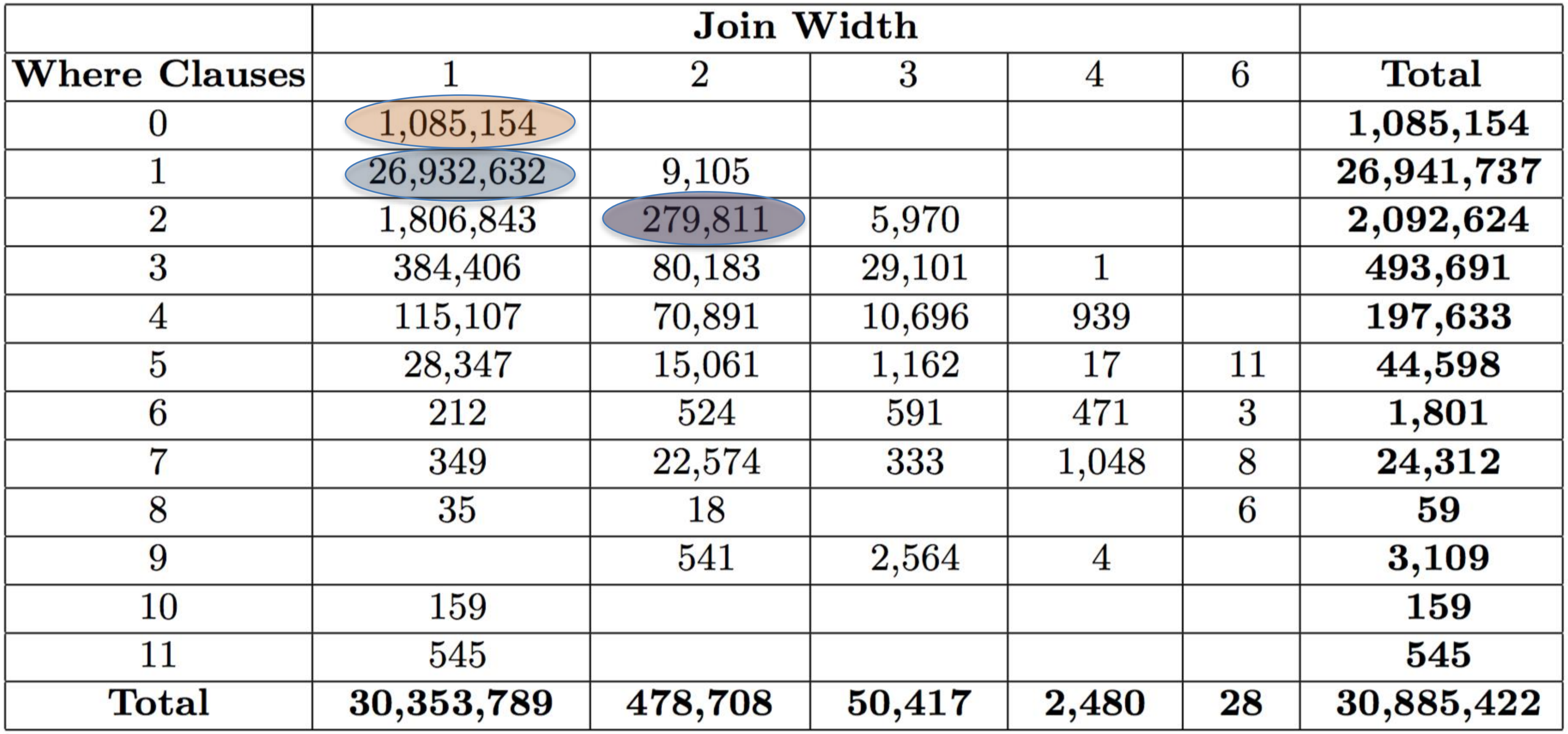




- 86% of all queries are simple single table scans/look-ups.
- Extreme – ‘Google Play Services’ queries accessing 8 distinct tables.

Reference: http://odin.cse.buffalo.edu/wp-content/uploads/2015/06/2015-TPCTC-SQLite-submitted.pdf

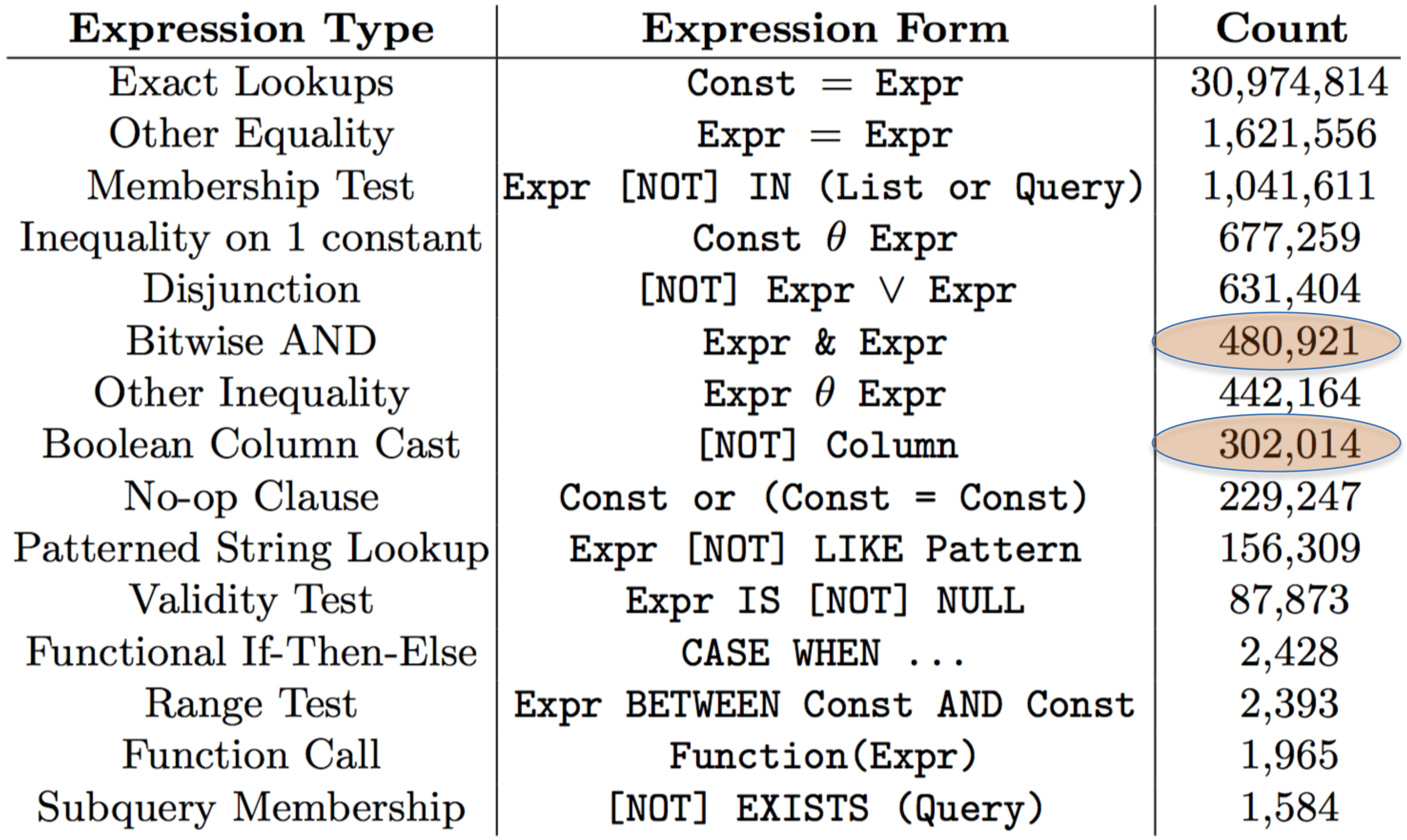
Observations from ‘Pocket Data: The Need for TPC-MOBILE’ paper



	Join Width					
Where Clauses	1	2	3	4	6	**Total**
0	1,085,154					**1,085,154**
1	26,932,632	9,105				**26,941,737**
2	1,806,843	279,811	5,970			**2,092,624**
3	384,406	80,183	29,101	1		**493,691**
4	115,107	70,891	10,696	939		**197,633**
5	28,347	15,061	1,162	17	11	**44,598**
6	212	524	591	471	3	**1,801**
7	349	22,574	333	1,048	8	**24,312**
8	35	18			6	**59**
9		541	2,564	4		**3,109**
10	159					**159**
11	545					**545**
Total	**30,353,789**	**478,708**	**50,417**	**2,480**	**28**	**30,885,422**

Reference: http://odin.cse.buffalo.edu/wp-content/uploads/2015/06/2015-TPCTC-SQLite-submitted.pdf

Observations from ‘Pocket Data: The Need for TPC-MOBILE’ paper



Expression Type	Expression Form	Count
Exact Lookups	`Const = Expr`	30,974,814
Other Equality	`Expr = Expr`	1,621,556
Membership Test	`Expr [NOT] IN (List or Query)`	1,041,611
Inequality on 1 constant	`Const` θ `Expr`	677,259
Disjunction	`[NOT] Expr` $\vee$ `Expr`	631,404
Bitwise AND	`Expr & Expr`	480,921
Other Inequality	`Expr` θ `Expr`	442,164
Boolean Column Cast	`[NOT] Column`	302,014
No-op Clause	`Const or (Const = Const)`	229,247
Patterned String Lookup	`Expr [NOT] LIKE Pattern`	156,309
Validity Test	`Expr IS [NOT] NULL`	87,873
Functional If-Then-Else	`CASE WHEN ...`	2,428
Range Test	`Expr BETWEEN Const AND Const`	2,393
Function Call	`Function(Expr)`	1,965
Subquery Membership	`[NOT] EXISTS (Query)`	1,584

WHERE clause expression structures, and the number of SELECT queries in which the structure appears as a conjunctive clause.

Reference: http://odin.cse.buffalo.edu/wp-content/uploads/2015/06/2015-TPCTC-SQLite-submitted.pdf

Observations from ‘Pocket Data: The Need for TPC-MOBILE’ paper

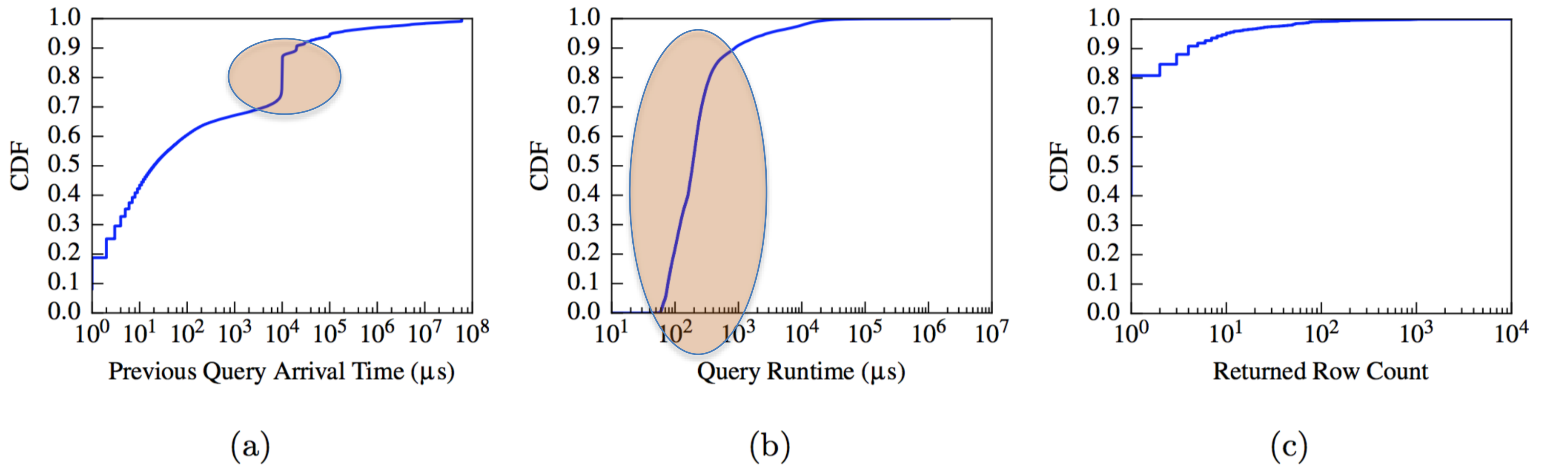


Fig. 12: **Summary Statistics for Android SQLite Queries. Distributions of (a) inter-query arrival times, (b) query runtimes, and (c) rows returned per query.**

- 20% queries periodic (File Locks?)
- 85% queries run in 0.1ms
- 80% queries returned single row (key-value lookup)

Reference: http://odin.cse.buffalo.edu/wp-content/uploads/2015/06/2015-TPCTC-SQLite-submitted.pdf

Ideation

- Analysis per Application (Read % , Read/Write ratio)
  - Long tail distribution skews results.
- Cluster based analysis
  - Analyze patterns within cluster
  - Generalize the behavior
  - Explain the behavior
  - How certain that a new app of this cluster will behave same?

Ideation

- Cluster Analysis [Contd..]
  - Finding similar clusters for each feature. Combine them into one if they behave same.
  - Split a cluster into two if there are two sets of query access patterns and they can be explained.
  - Frequency of app usage within cluster should not demand different benchmarks.
    - It should be driven by scale factor and burst factors?

Steps ahead

- Identifying and finalizing the right features
- Phone data log file extraction.
- Implement the ideas.
  - Per app basis analysis & cluster based analysis