

Hello!

CS 744: SNOWFLAKE

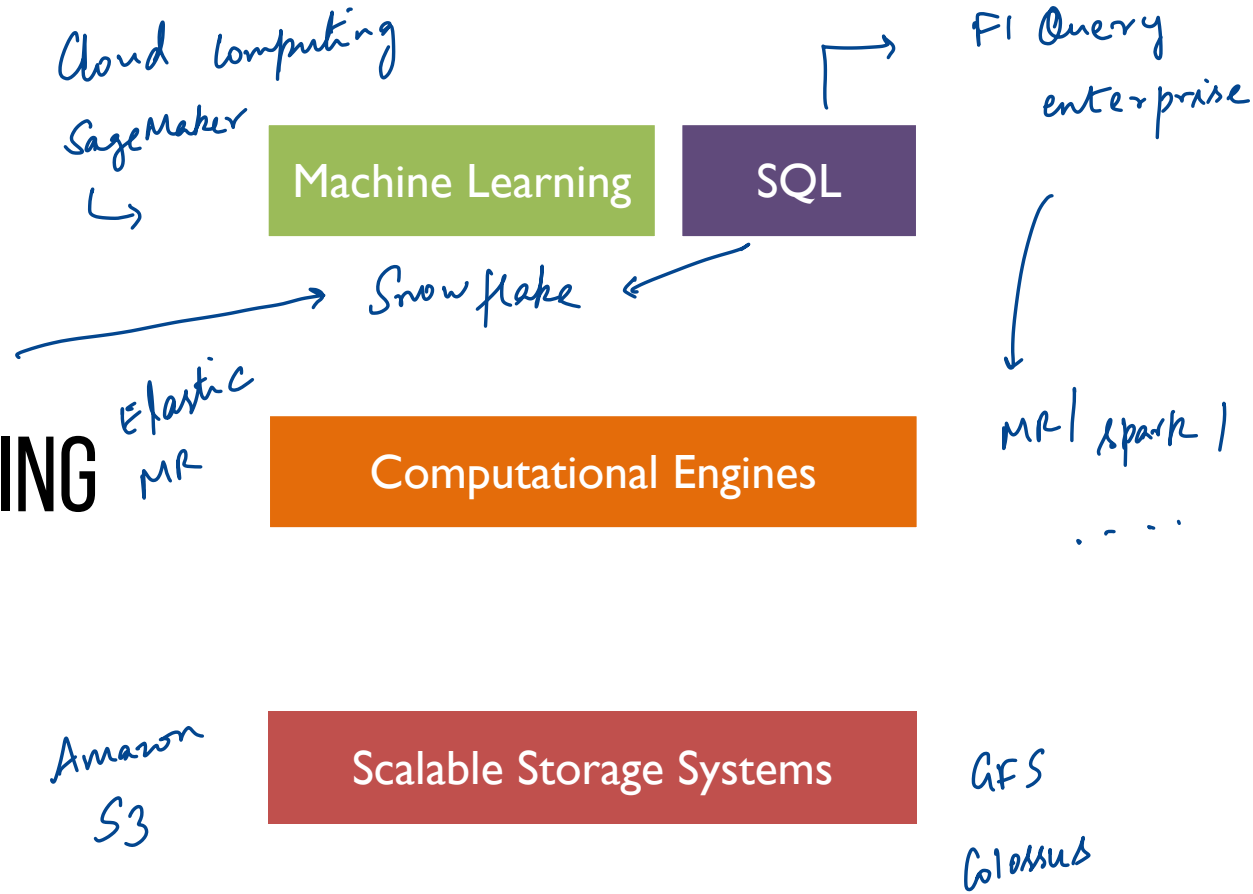
Shivaram Venkataraman

Spring 2025

ADMINISTRIVIA

- Midterm on Tuesday! → 1 pm in this room
- Assignment 2 grading → soon!
- Project proposal feedback
↳ next week

CLOUD COMPUTING STACK

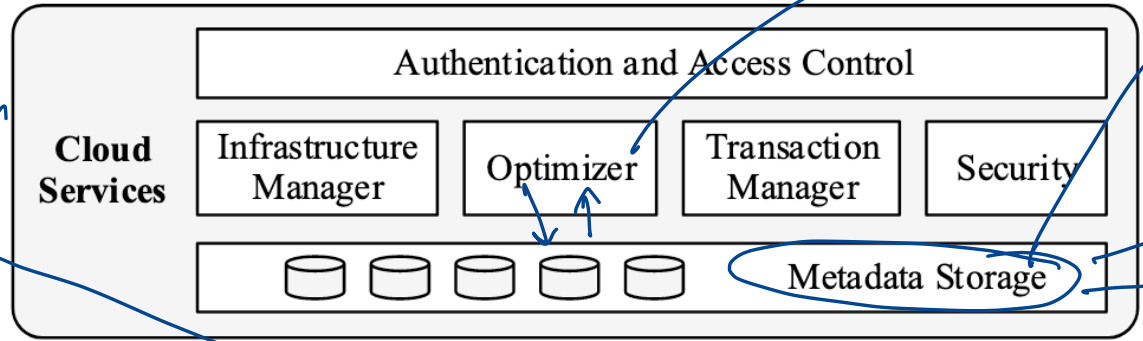


SNOWFLAKE: GOALS



SNOWFLAKE DESIGN

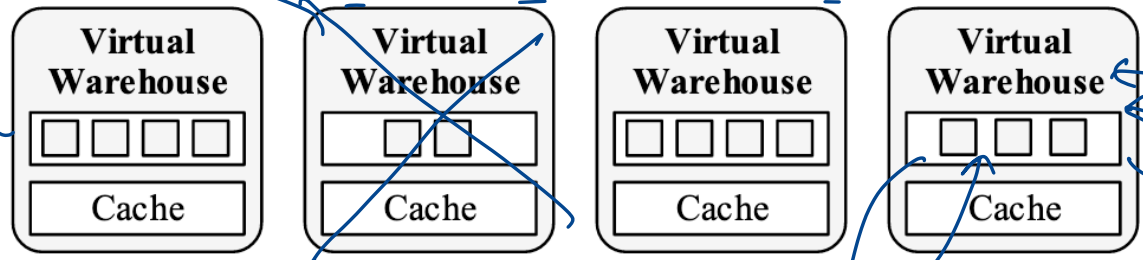
Control plane
→ Coordination
stateless



read schema
which tables are where

externalize
Redis / Mongo DB

Data plane
set of VMs
insert data

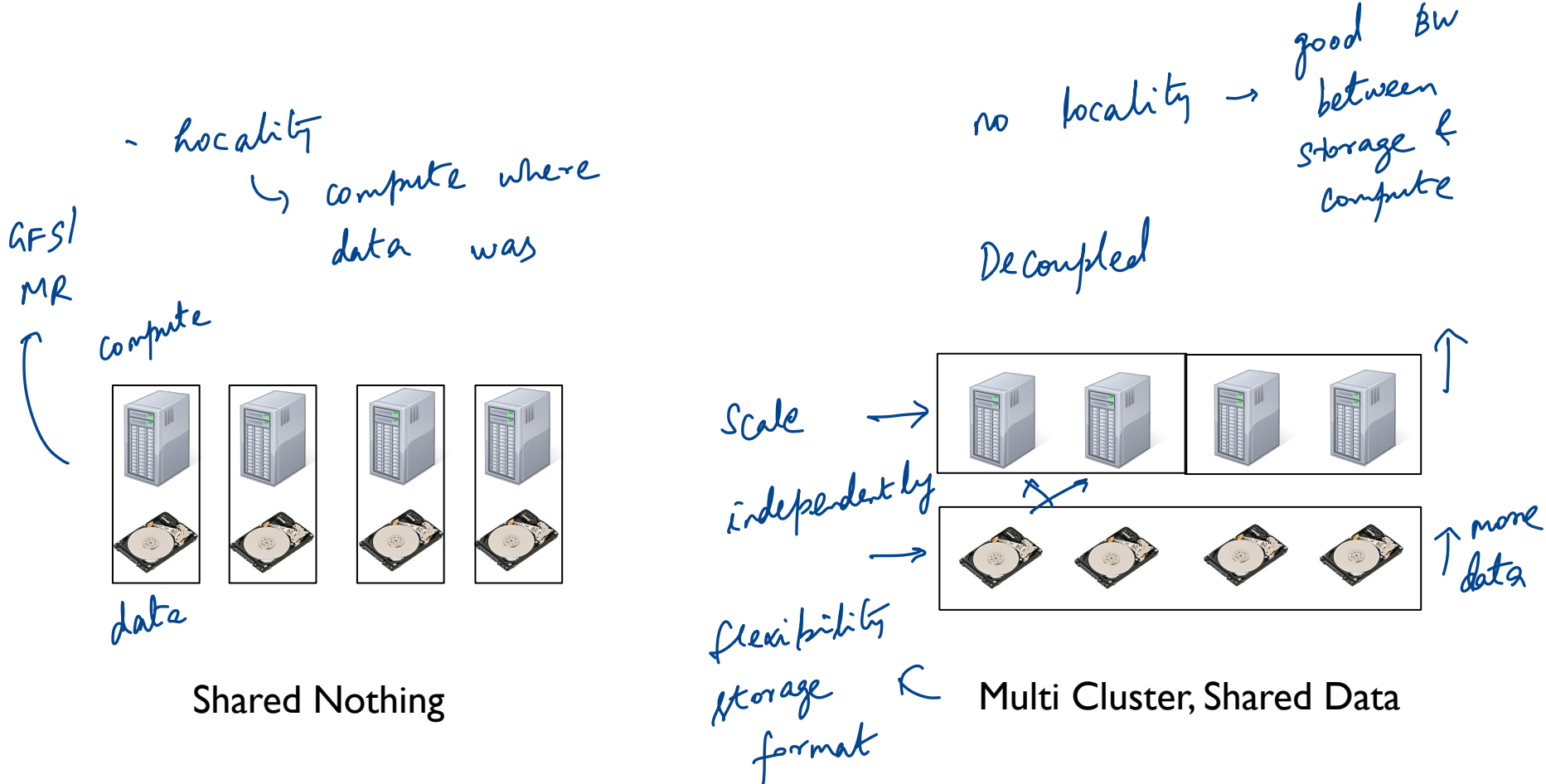


each user launches vw



S3 buckets globally accessible

STORAGE VS COMPUTE



STORAGE: HYBRID COLUMNAR

Updates

create new chunk

Alice	32
Bob	22
Eve	24
Victor	27

Columnar format

encoding

skip bytes for queries

store columnar format

split

by rows

2 rows

row1 row2
Alice,32,Bob,22

row3 row4
Eve,24,Victor,27

Row-oriented

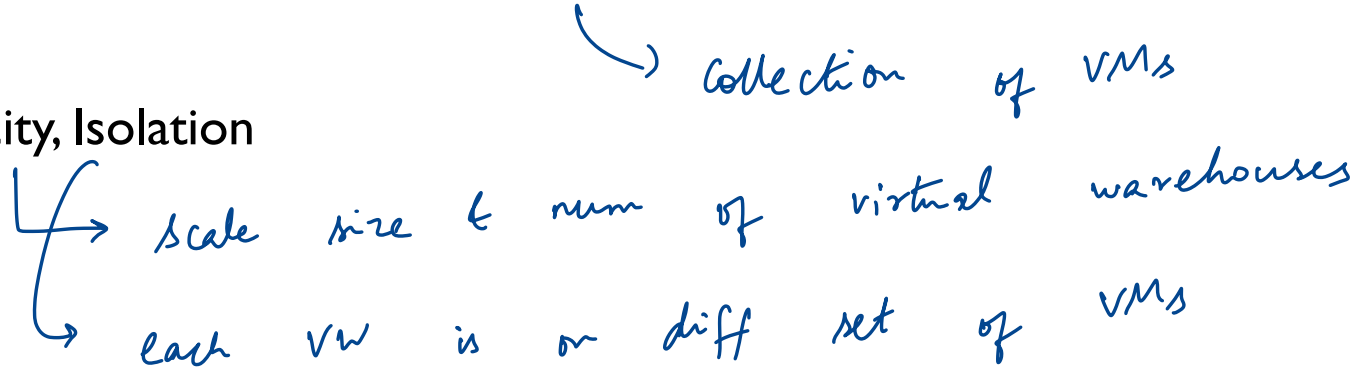
Alice, Bob, 32, 22

Eve, Victor, 24, 27

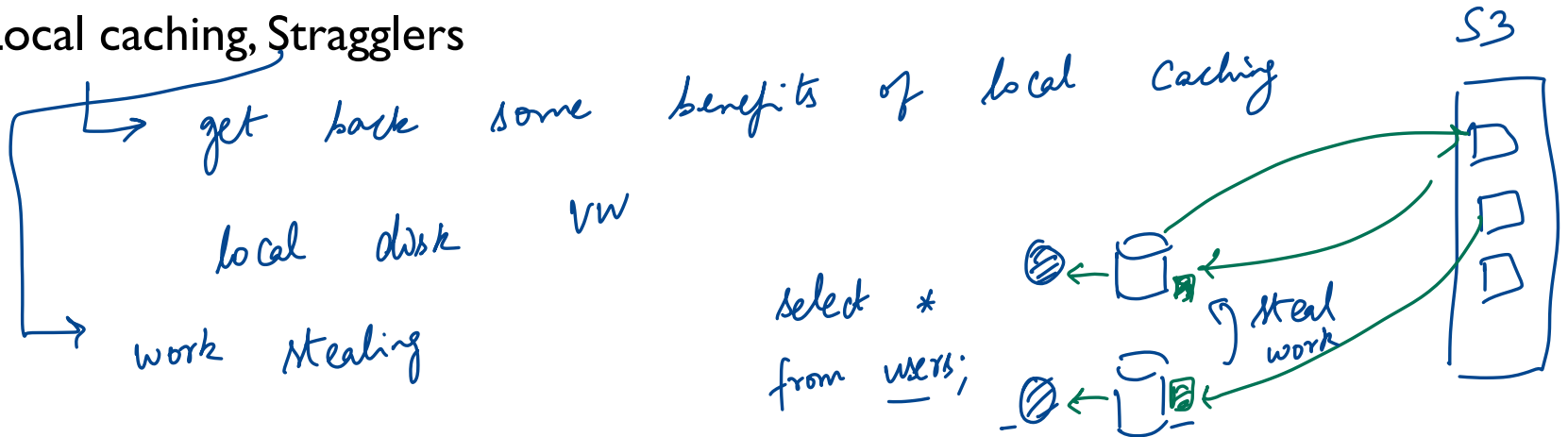
Hybrid Columnar

VIRTUAL WAREHOUSES

Elasticity, Isolation

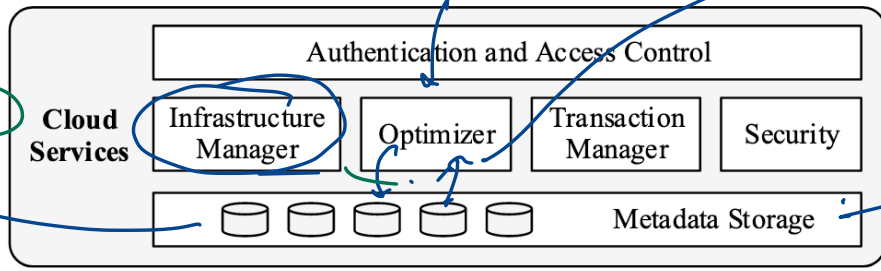


Local caching, Stragglers



CLOUD SERVICES

chunk -v0 = s3 file
-v1 : s3 file
-v2 : s3 file



latest version of each chunk

separate storage system

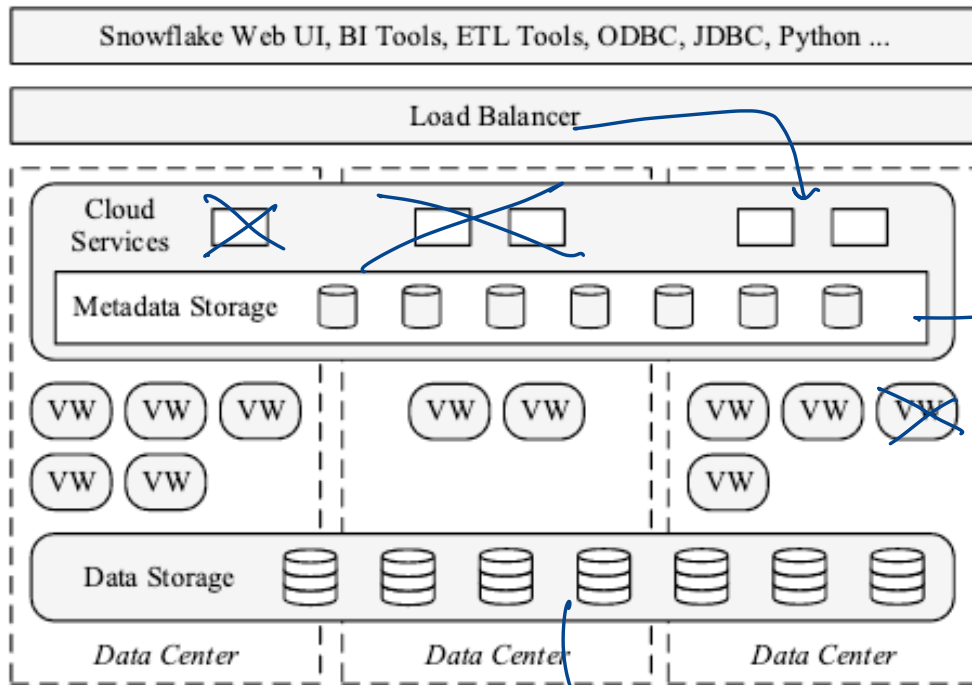
Concurrency Control

↳ How do you do updates & queries at same time?
↳ every chunk of table
↳ multiple versions
↳ query is tied to a version of each partition / chunk

Pruning

↳ which partitions need to scanned
↳ statistics prune the partitions based on query

FAULT TOLERANCE



stateless services
restart a new vw
if you have a failure

Always On → replicate across data centers

On Demand

Infinite

replication if you want survive DC failure

SEMI STRUCTURED DATA

```
{  
  first_name: "john",  
  last_name: "doe",  
  order id: "1234",  
}  
{  
  first_name: "bucky",  
  last_name: "badger",  
  order_id: "52342",  
  order_date: "3/3/2020",  
}
```

Extraction, Flattening operations

row. order_id integer

type inference

Infer types, Pruning

TIME TRAVEL?

```
SELECT * FROM my_table [AT(TIMESTAMP =>
    'Mon, 01 May 2015 16:20:00 -0700'::timestamp)];
SELECT * FROM my_table AT(OFFSET => -60*5); -- 5 min ago
SELECT * FROM my_table BEFORE(STATEMENT =>
    '8e5d0ca9-005e-44e6-b858-a8f5b37c5726');
```

*partitions as of
this date*

Multiple versions of table (MVCC)

Undo accidental deletes

Cheap to clone / snapshot a table



*Save a version of
table as diff name*

SUMMARY, TAKEAWAYS

Snowflake

- Cloud computing → Elastic data warehouse
- Key idea: Separation of compute and storage!

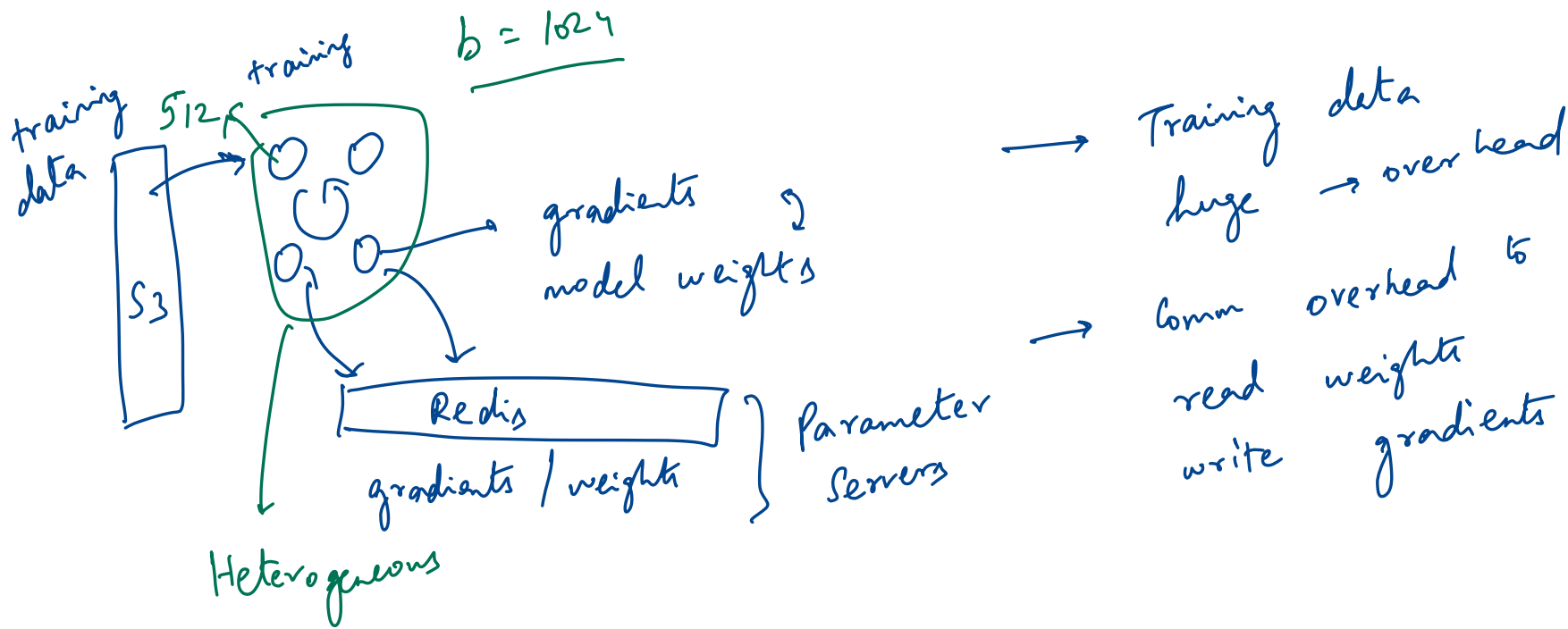
- Hybrid columnar storage format
- Elastic compute with virtual warehouses
- Pruning, semi-structured optimizations, fault tolerant

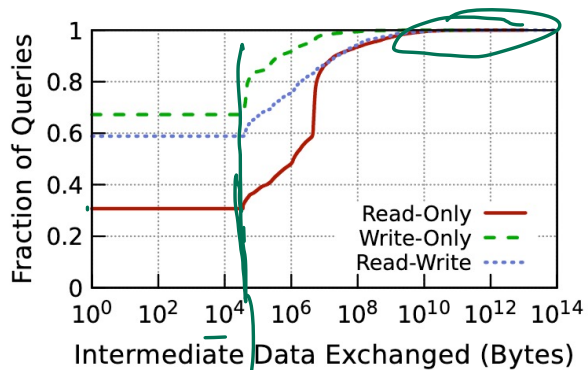


DISCUSSION

<https://forms.gle/KjsqAajFrVyyMi4s8>

We see how Snowflake leads to the design of an elastic data warehouse. If we were to similarly design an Elastic PyTorch for training how would the design look? What are some design trade-offs compared to existing PyTorch?

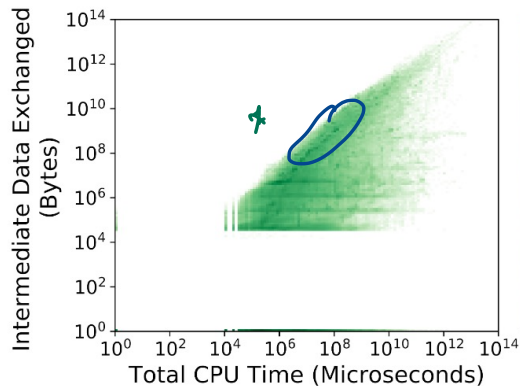




Analysis of Snowflake queries

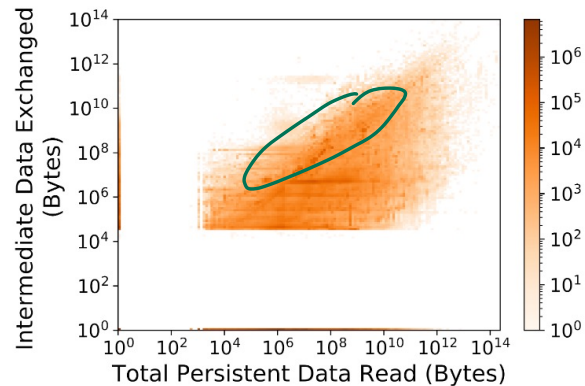
32 000 bytes

~ 32 KB



queries

Correlation between



→ S3

intermediate vs. CPU time

NEXT STEPS

Next class: Midterm!