DataStax

REST, GraphQL and Document API SAI indexing at scale

LEVEL UP with the DataStax Developers

Partnering to Deliver Transformational Outcomes

Logistics and Asset Management

DELTA FedEx Booking.com

Inventory and Catalog Management

Lowe's macy's The Home Depot Walgreens

Real Time Payments

PayPal venmo

Customer 360

HBO verizon The Walt Disney Company

Fraud Detection and Prevention

ACI Universal Payments Bank of America Capital One

Performance Management

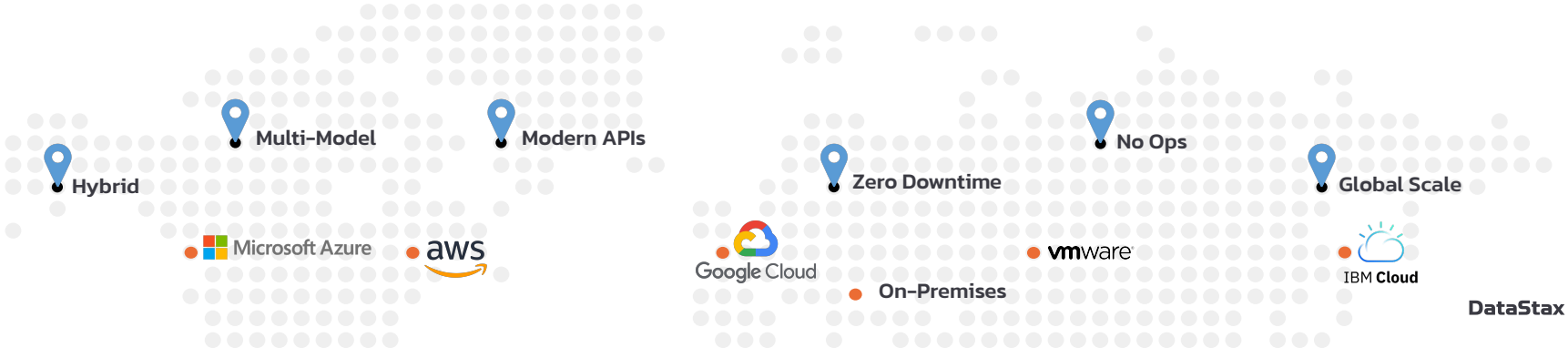
EQUINIX Constant Contact

Supply Chain Management

DELL elementum SAP Ariba

Digital Delivery

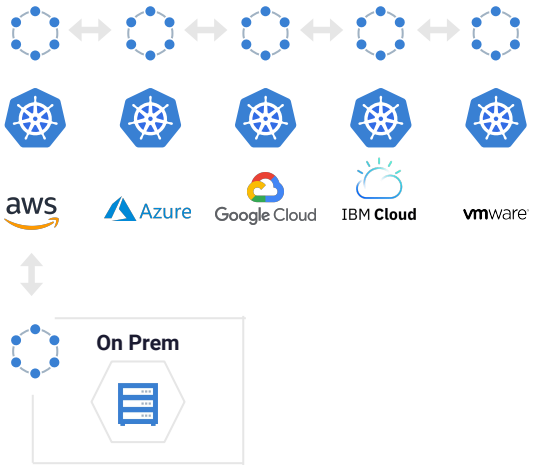
NETFLIX sky sling TELEVISION



Cassandra Data Platform

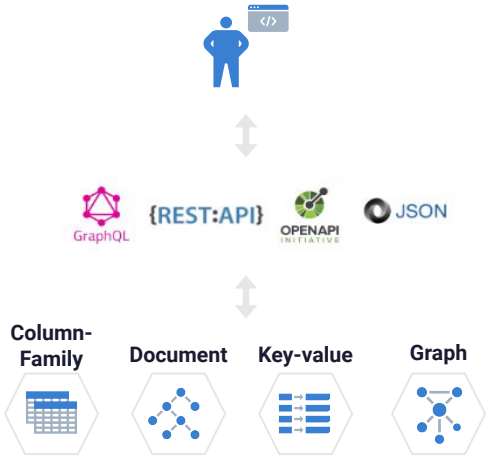
Hybrid

Operate at scale, anywhere



Database

Any API, Any Data Model



As-a-Service

Managed

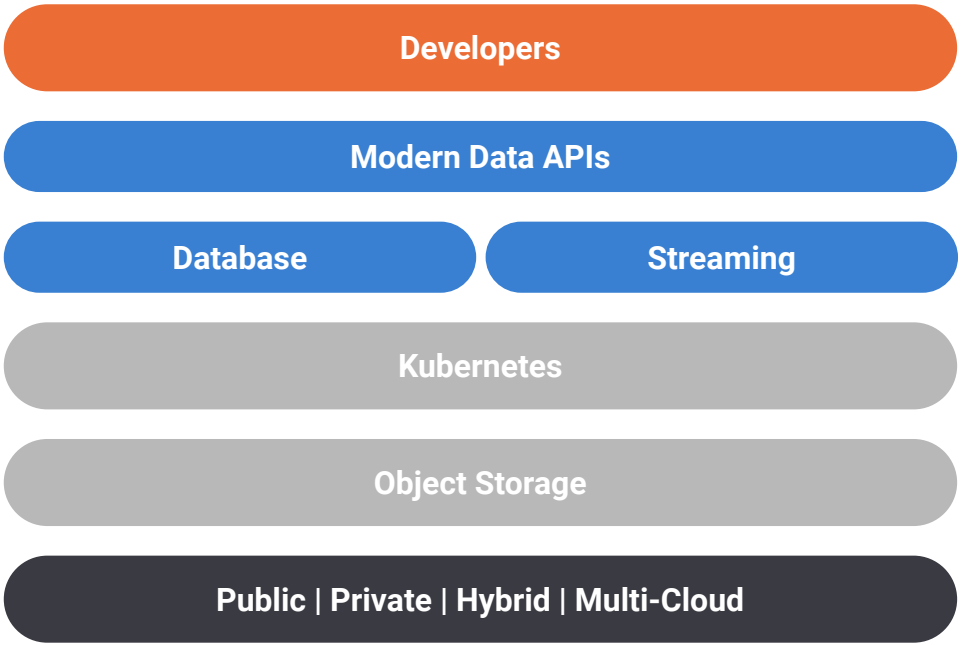
DataStax Astra

Self Managed

DataStax Enterprise

Apache CASSANDRA + DataStax Luna

Modern Data Apps Require an Open Data Stack



Developer Ready.
Designed for the new apps of tomorrow.

API adoption CAGR ~20% 2020-2024

Microservices adoption CAGR 21.6%

Kubernetes-Based.
Transforming the enterprise stack.

Enterprises adoption is 48% in 2020 going to 85% by 2025

80% of ISVs will rebuild their stacks for containers by 2025

Cloud-Delivered.

47% of businesses write applications specifically leveraging underlying cloud infrastructure

Serverless and Function-as-service has 75% annual growth rate

Astra

Serverless Managed Cassandra

Cut your cost by half!

What is Serverless

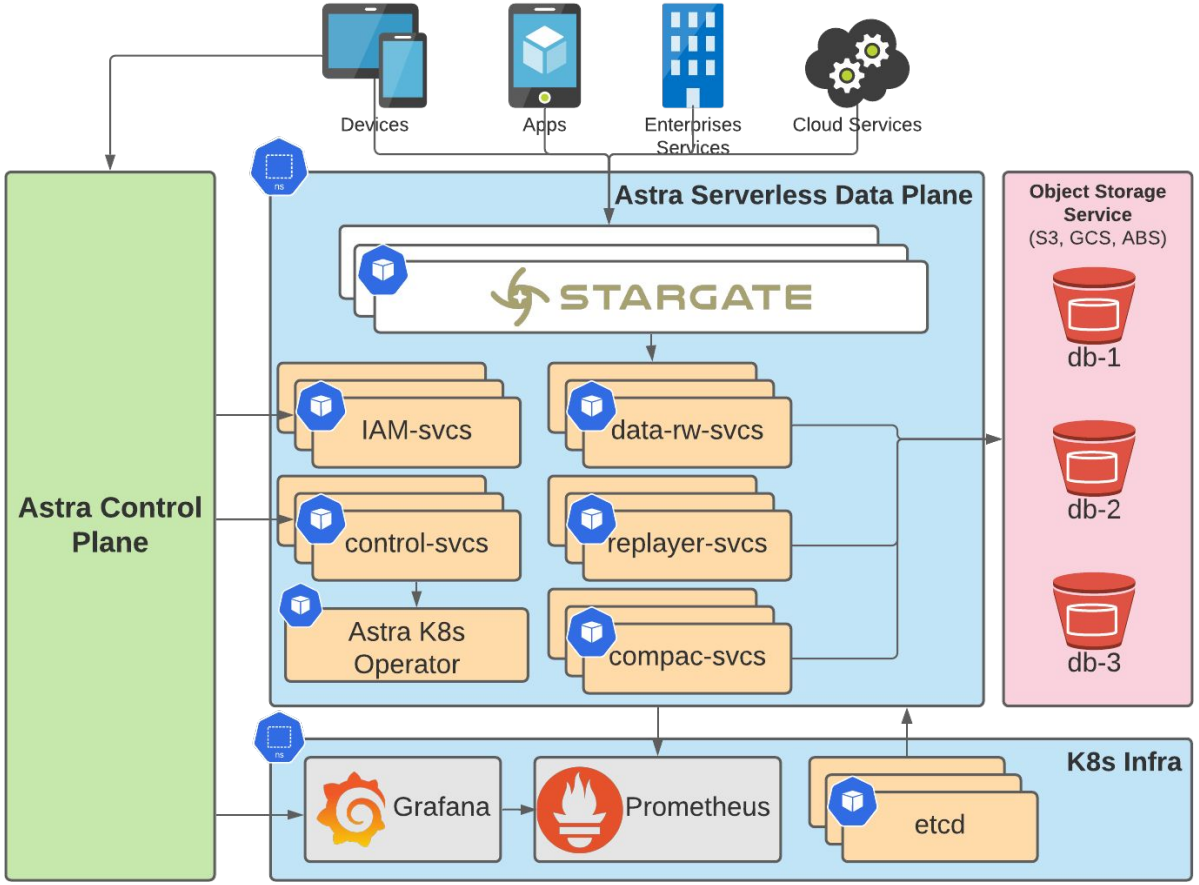
- Cloud provider responsible for code execution
- Cloud provider dynamically allocates resources
- Charging only for resources used

Astra

Serverless Cassandra as-a Service

- **Overprovisioning Solution**: Elasticity makes sizing for the peak a thing of the past.
- **Reducing the Silos**: Knowing that workloads will not affect each other, enable and even encourage people to avoid duplicating the data across the organisations.
- **No idle or abandoned instances**: Astra charging model is PAYG, meaning that there is no charge if there is no usage. This enables developers and QA to build resilient apps cost effectively.

How we broke the Cassandra Monolith and made it Serverless



Unit of Measure - Executed DB Transactions and Consumed Storage

Apps are about Read and Write
Infrastructure is all the storage

Write Requests
API calls to write data to your table are billed in write request units

Reads Requests
API calls to read data from your table are billed in read request units

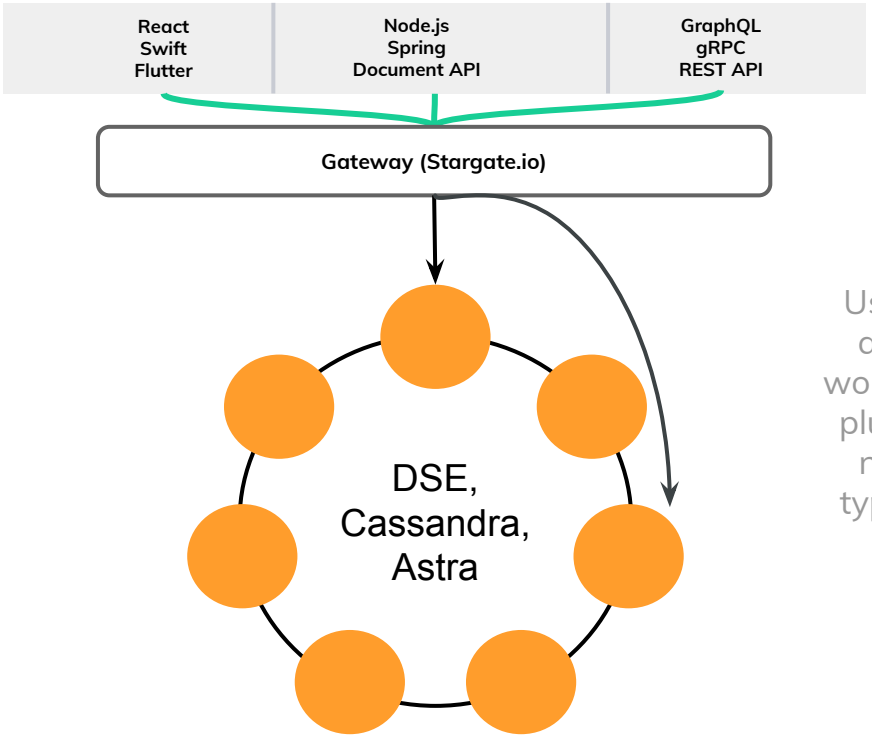
Data Storage
Billable data is charged per GB

Stargate Data gateway

REST, GraphQL and Document API

Developer Efficiency!

Stargate : Super Charge Cassandra



Use database for any application workload by adding plugin support for new APIs, data types, and access methods

Connection Gateway

Decouple Application from Database Release cycles

Security

Accelerate Development

Familiar playing ground for Developers

Rest API

JSON / Document API - Schemaless

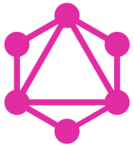
GraphQL API

{REST}

REST API

Greater flexibility and Faster development

- **Authentication**
  - Exposes Authentication API for token generation
  - Secures every single Request/Response to Astra
- **Schema**
  - The Schema API allows you to interact with keyspaces and tables in your database
  - /api/rest/v2/schemas/keyspaces/{keyspace-id}/tables
- **Data**
  - The Data API allows you to add, update, read and delete rows in your database
  - /api/rest/v2/keyspaces/{keyspace-id}/{table-id}



GraphQL API

Make queries fast, flexible and client-friendly

- Developer can pick exact data the client UI needs
- Reduce number of queries by retrieving all relevant data from a single endpoint
- GraphQL objects generated for every table
  - Queries - Read data
  - Mutations - Insert and Modify data
- Schema API
  - For DDL operations (create, drop Table)
  - apps.astra.datastax.com/api/graphql-schema
- Query API
  - Querying and modifying table data using Graph fields
  - apps.astra.datastax.com/api/graphql/{keyspace_name}

Document API

Save and search schemaless JSON documents without data modeling

- Store JSON documents and automatically create schema
- Automatic SAI Indexes for flexible search on the JSON documents
- Fetch Full-Document or Sub-Document
- Update a section of the document without reading the entire document
- Fetch all documents in the collection
- Search across the entire collection

Let get started

Using Stargate on Astra

https://github.com/michelderu/datastax-workshop

Storage Attached Indexes

Ad-Hoc Querying

Developer Efficiency!

What's wrong with the following...

CQL Copy

```cql
CREATE TABLE demo.personnel (id int,
firstname text,
lastname text,
age int,
employee_start_date date,
PRIMARY KEY (id)
);
```

CQL Copy

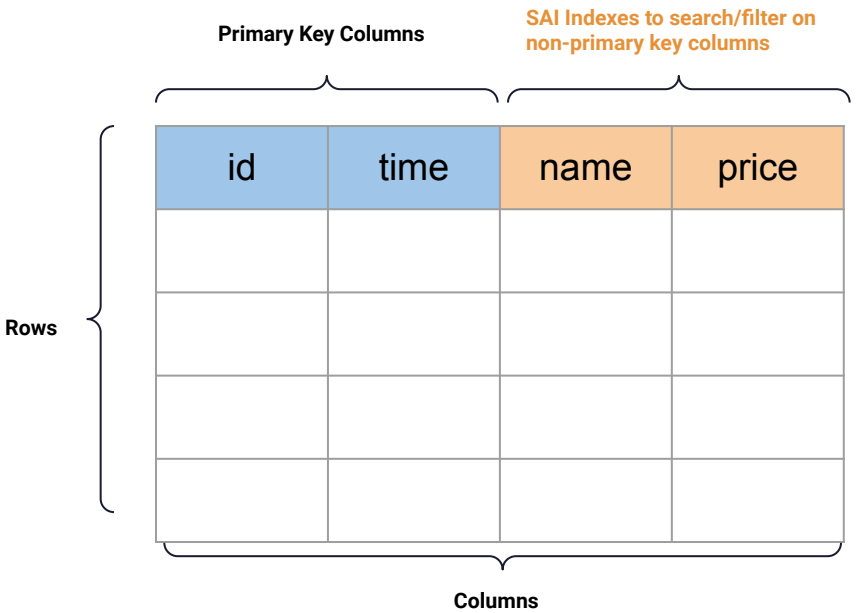
```cql
SELECT firstname, lastname
FROM demo.personnel
WHERE age > 30
```

Storage Attached Indexes

SAI provides traditional, relational database style indexing and querying capabilities for Cassandra which is easier to use, more efficient and simpler to maintain

- Query Data using non Primary-Key Columns
- Eliminates the need to use ALLOW FILTERING keyword or create custom tables for each query pattern
- The indexes live where data lives in Cassandra (on disk - SSTable & in memory - Memtable)
- Column-based for flexibility
- Minimal user configuration
- Not a Full-text Search

When to use SAI



SAI Indexes allow you to query columns outside the Cassandra partition key without using the ALLOW FILTERING keyword or creating custom tables for each query pattern

Example:
*SELECT * FROM foo* ***WHERE name='CPU' AND price = 3000***

Pain points solved:

- Code around restrictive C* read path
- Duplicating denormalized data to query non PK fields
- The painful process of learning about all the edge cases with 2is
- Having to wait for DSE Search or other Search tools to be integrated with C*

SAI : Indexing & Querying

Indexing

Primary Key Columns

SAI Indexes to search/filter on non-primary key columns

id	time	name	price

```
CREATE CUSTOM INDEX name_idx ON products(name)
USING 'StorageAttachedIndex'
WITH OPTIONS = {'case_sensitive':false, 'normalize':true};
```

```
CREATE CUSTOM INDEX price_idx ON products(price)
USING 'StorageAttachedIndex'
```

Querying

Primary Key Columns

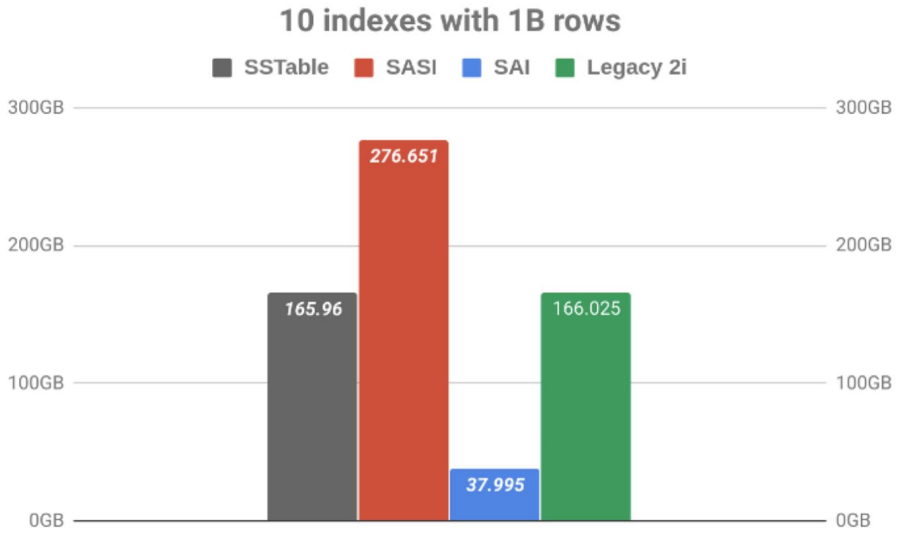
id	time	name	price
567	11:42:35	'Basic Task CPU'	3000
958	10:22:43	'Robotic Arms'	7000

```
SELECT * FROM products WHERE name = 'Basic Tasks CPU'
```

```
SELECT * FROM products WHERE price = 7000
```

Storage attached indexes

Feature	Available Now
Query Operators	=, <, >, <=, >= (Numerics); CONTAINS, CONTAINS Key, CONTAINS VALUE, = (Strings)
Apache Cassandra Types	ASCII, BIGINT, DATE, DECIMAL, DOUBLE, FLOAT, INT, INET SMALLINT, TEXT, TIME, TIMESTAMP, TIMEUUID, TINYINT, UUID, VARCHAR, VARINT, Collection types
Fields Available for Indexing	Primary Key and Non Key Fields



Let get started

Using Stargate on Astra

https://github.com/michelderu/datastax-workshop

Further reading

https://www.datastax.com/dev/cassandra-indexing

https://www.datastax.com/blog/eliminate-trade-offs-between-database-ease-use-and-massive-scale-sai-storage-attached

https://www.datastax.com/dev/cassandra-indexing