DigitalSkola
Uncover The World of Digital Skills with Us

LEARNING PROGRESS REVIEW "WEEK 4"

BY : DREAM (KELOMPOK 1)

Data Realm Engineers And Maestros

ANGGOTA KELOMPOK

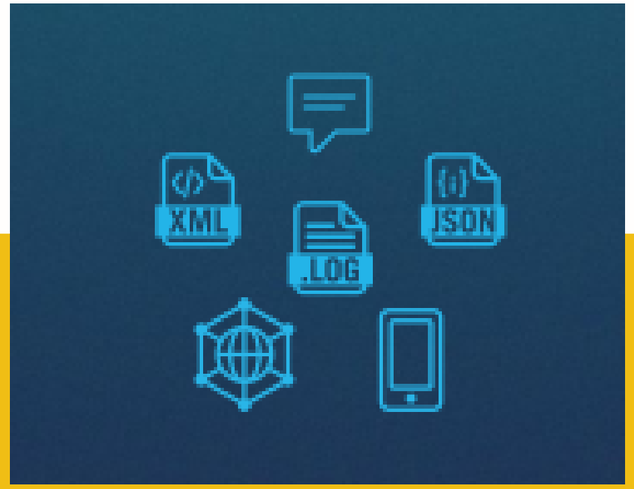
- Afroh Fauziah
- Althaf Taqiyyah
- Andi Rosilala
- Andrew Bintang Pratama
- Andrew Suadnya

DigitalSkola
Uncover The World of Digital Skills with Us

snowflake

SNOWFLAKE INTRODUCTION

TEKNOLOGI BARU MENGUBAH BAGAIMANA KITA MENGGUNAKAN DATA



Diversification of Analytics

Analytics is growing in importance, everywhere, and for everyone

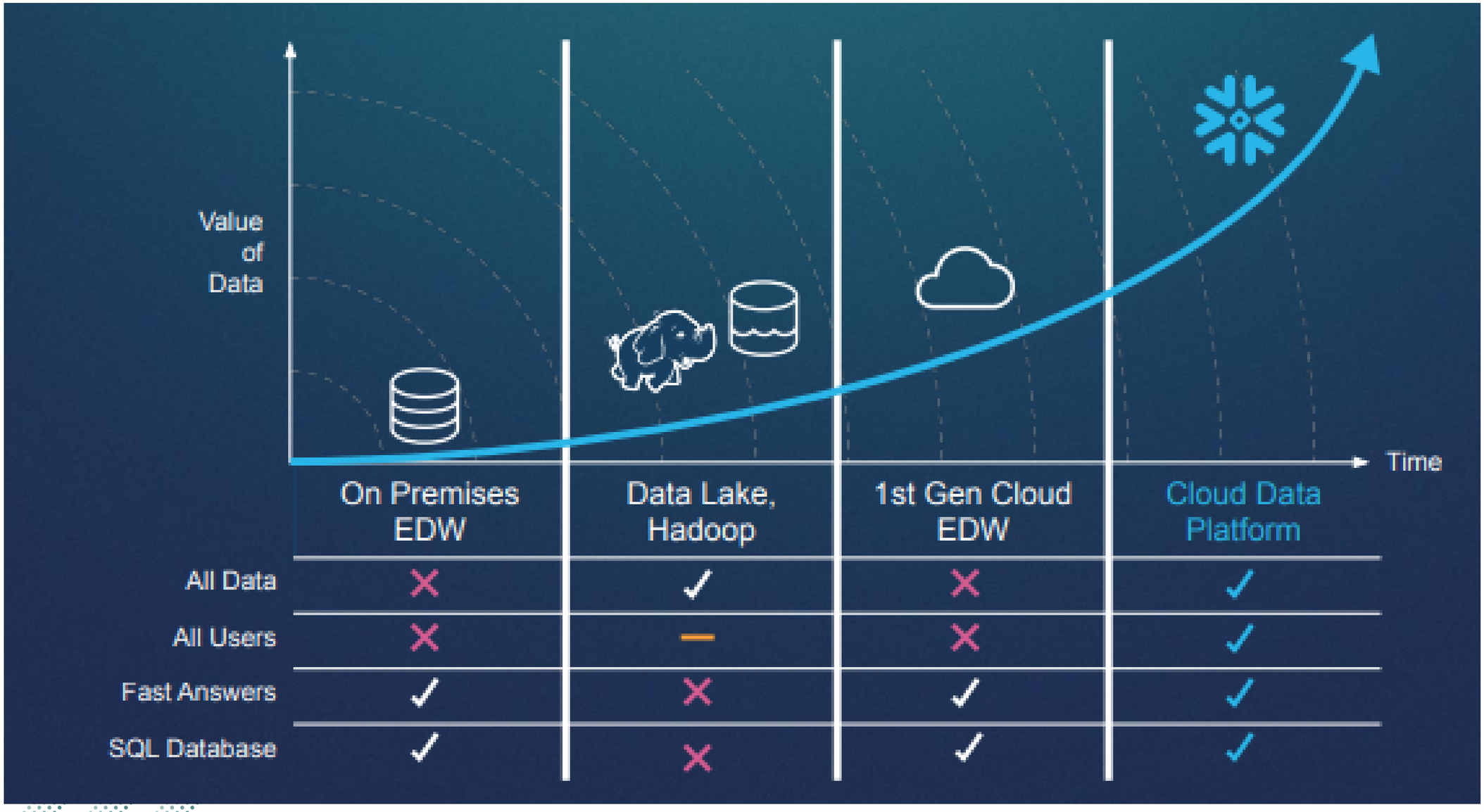
Explosion of Data

IoT, mobile, and social open up new opportunities for insight

Rise of the Cloud

Cloud gives us the ability to scale and centralize data

JOURNEY TO A CLOUD DATA PLATFORM



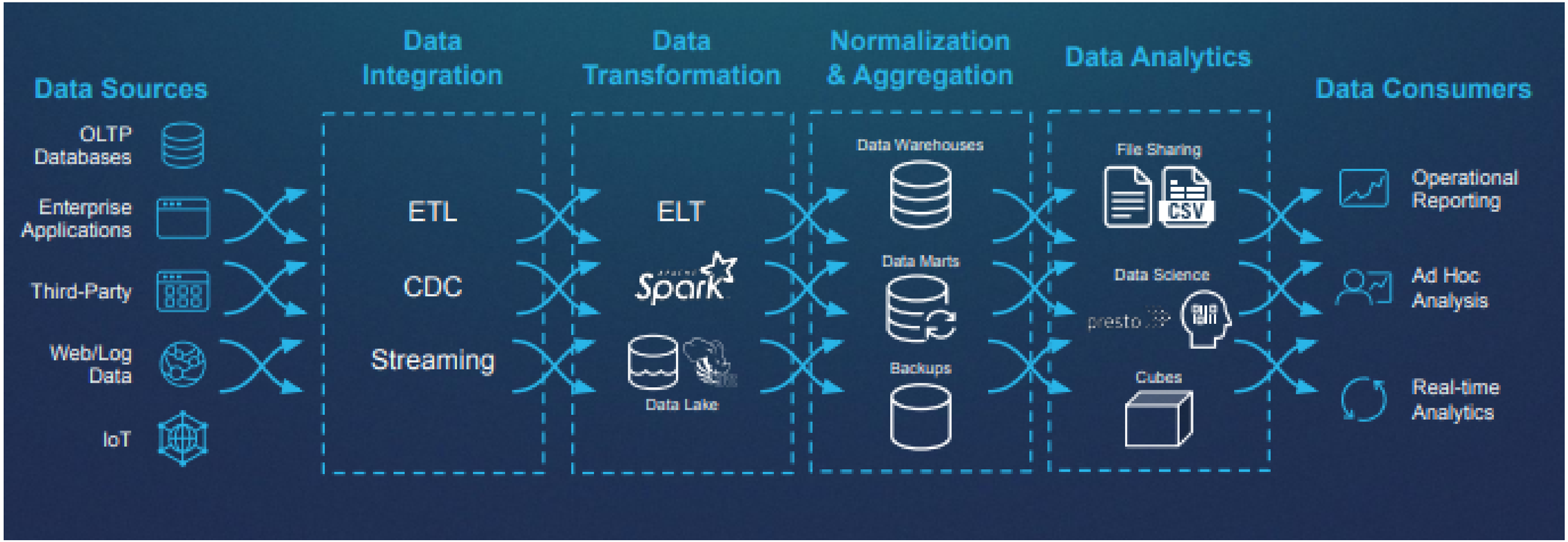
INTRODUCTION TO snowflake

> Suatu product / merk dari sebuah Analytic Data Warehouse sebagai Software As A Service (SAAS).

> Sepenuhnya running on cloud infrastructure beserta semua komponennya yang dimana snowflake tidak dapat berjalan pada infrastructure on premise.

- **Tidak ada hardware**
- **Tidak perlu install software, bisa di browser saja**
- **Maintenance, Management, dan tuning di handle oleh Snowflake**

TRADITIONAL DATA ARCHITECTURE



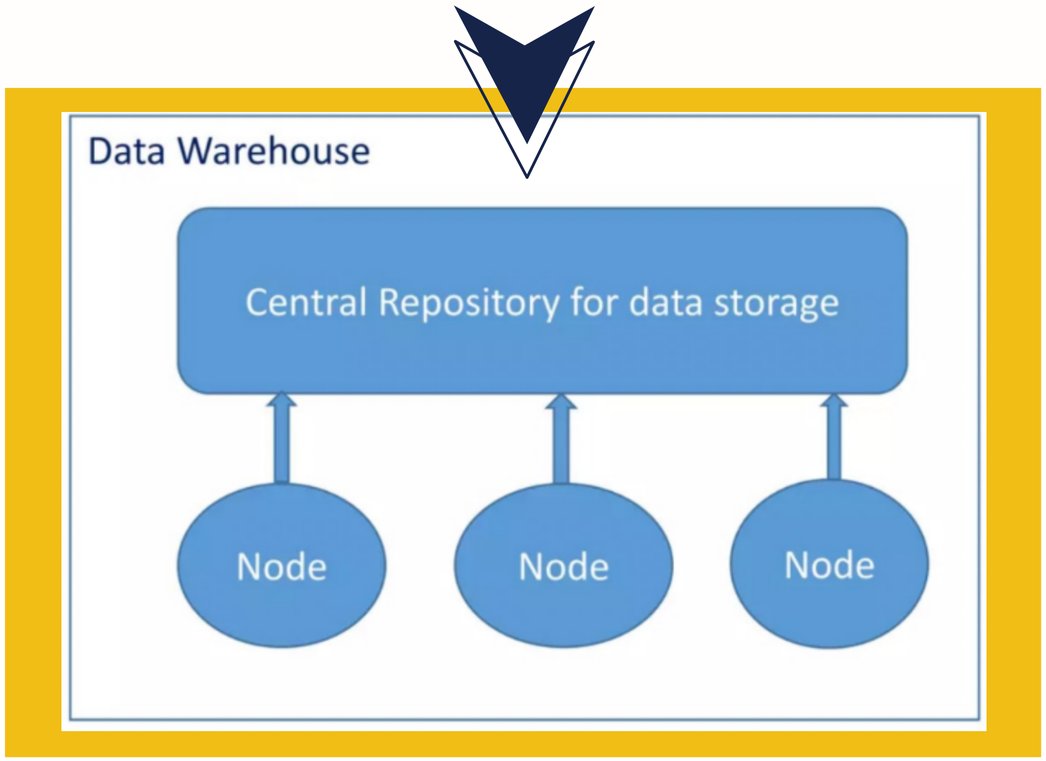
Complex and Costly with Multiple Copies of Data

DigitalSkola

SNOWFLAKE ARCHITECTURE

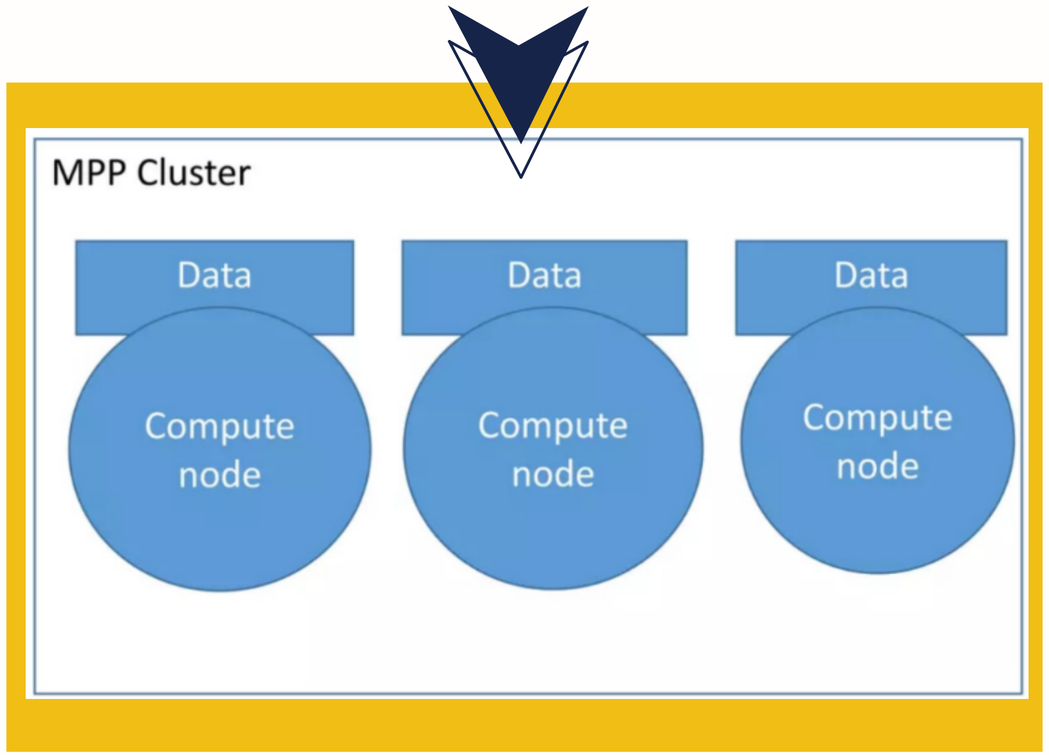
Shared Disk Database Architecture

penyimpanan / repository tersentralisasi, dapat diakses banyak node



Shared Nothing Database Architecture

superkomputer, dapat sewa node terpisah yg baru, jadi datanya ada di node tersebut

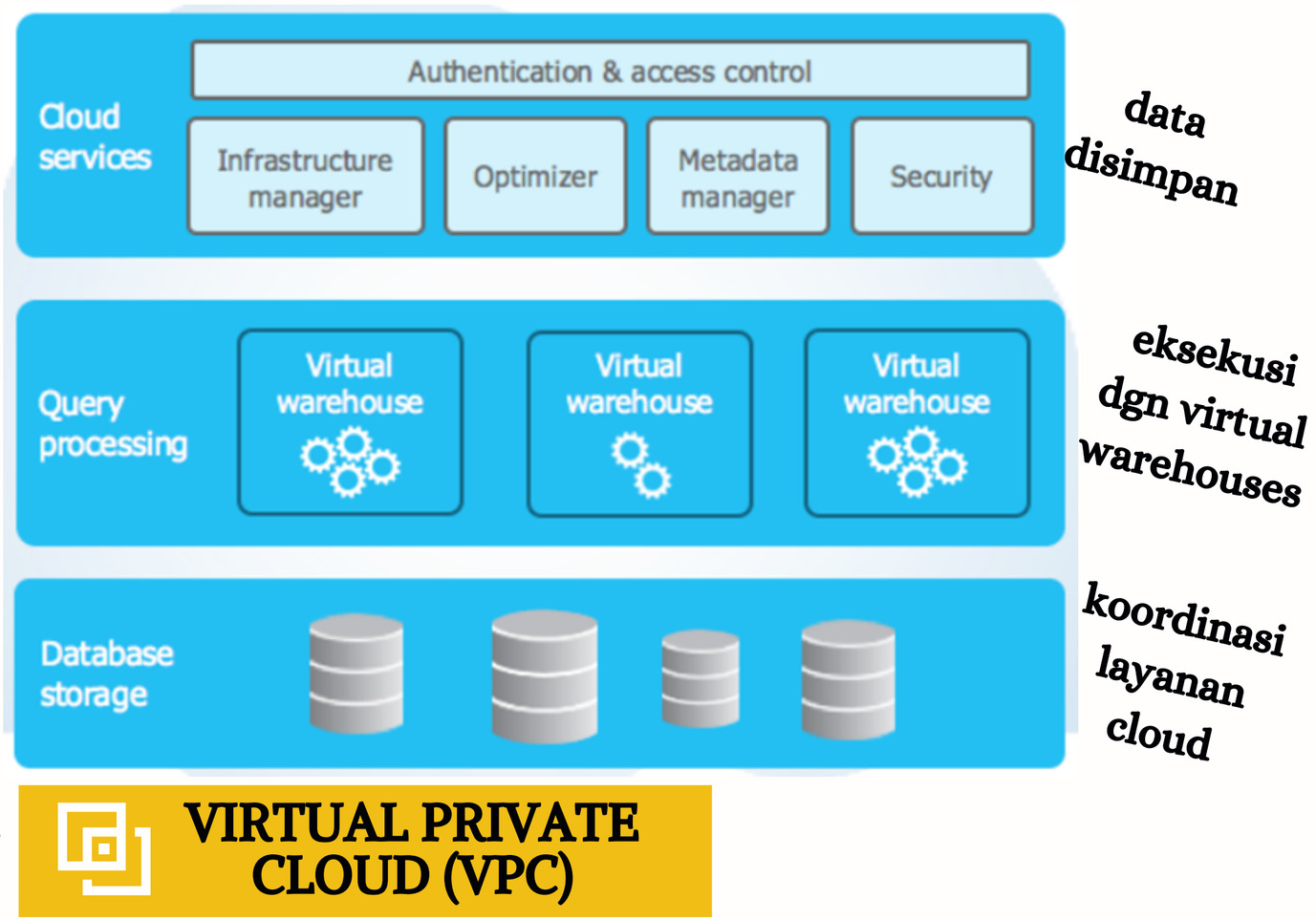


SNOWFLAKE ARCHITECTURE

Layer

- Database Layer
- Query Processing Layer
- Cloud Services

jaringan virtual yang lebih kecil daripada internet, dapat dioperasikan di pusat data sendiri, bisa beda zona



data disimpan

eksekusi dgn virtual warehouses

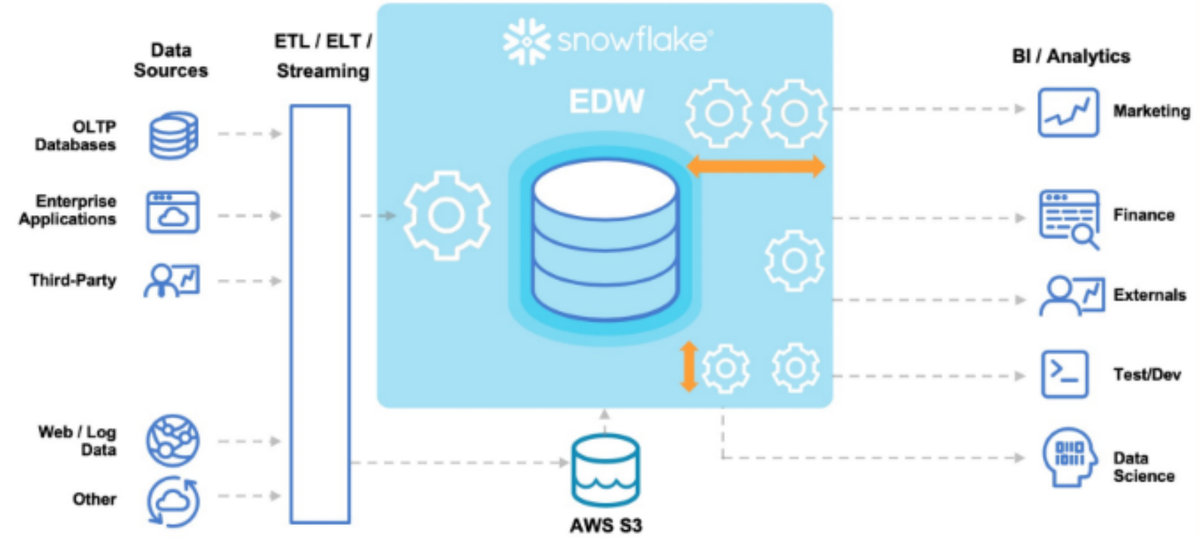
koordinasi layanan cloud

DigitalSkola

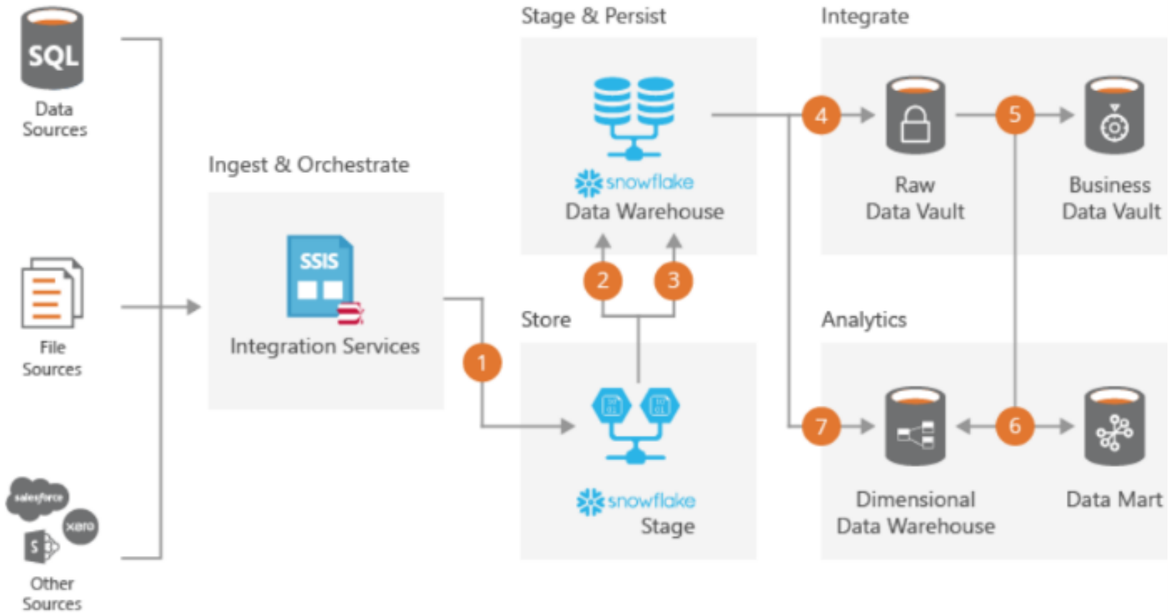
SNOWFLAKE BIG PICTURE ARCHITECTURE

perantara untuk mengelola berbagai sistem, modern data architecture

sebagai data warehouse, multi-cluster virtual warehouses

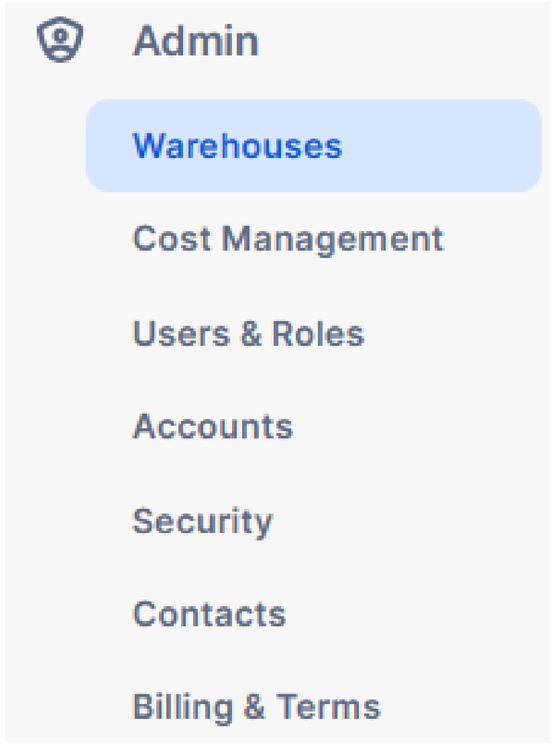


banyak kategorisasi / lini

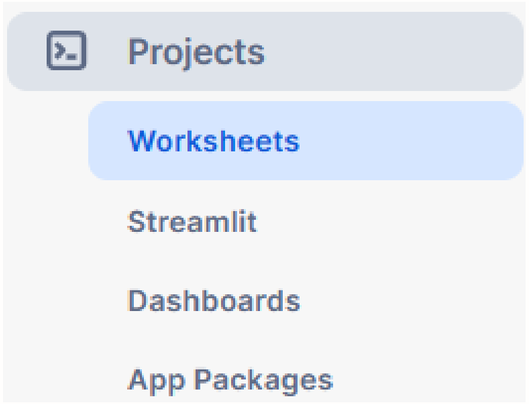


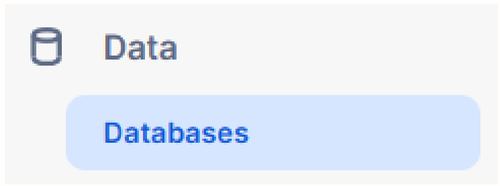
BAGIAN - BAGIAN SNOWFLAKE

memanjemen akun, akses, & pengaturan sistem

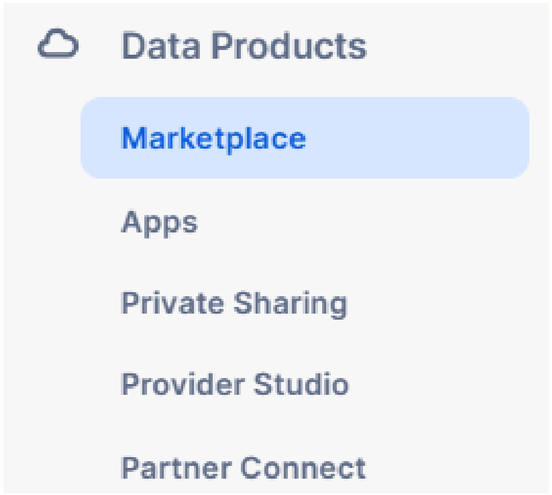


kumpulan proyek dan izinya



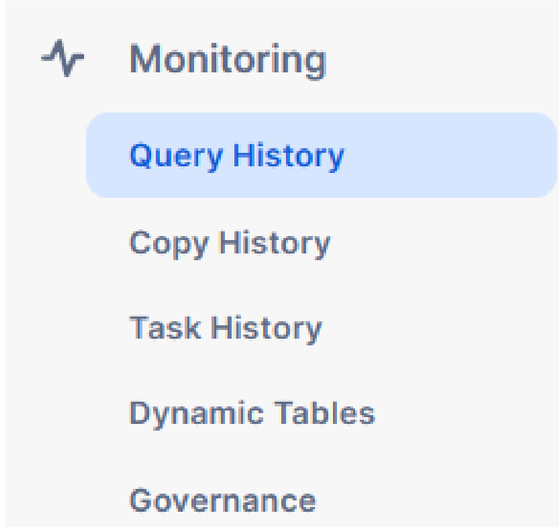


bentuk data skema, tabel, hasil data eksternal



pengembangan, pemrosesan, pemantauan

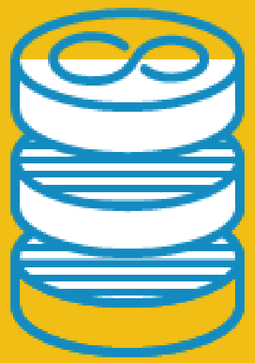
pemantauan kinerja, keamanan, operasi



DigitalSkola

HOW SNOWFLAKE IS DIFFERENT

SINGLE PLACE
FOR DATA



SCALE &
CONCURRENCY

ZERO
MANAGEMENT

INSTANT, LIVE
DATA SHARING

UNIQUE ARCHITECTURE:
MULTI-CLUSTER, SHARED DATA

DigitalSkola

TEST SNOWFLAKE

kueri lokasi file, registrasi

tasty bytes ke blog stgae

DigitalSkola

TEST SNOWFLAKE

mengcopy tabel dari query sebelumnya

```
78
79  ---> copy the Menu file into the Menu table
80  COPY INTO tasty_bytes_sample_data.raw_pos.menu
81  FROM @tasty_bytes_sample_data.public.blob_stage/raw_pos/menu/;
82
83
84  ----------------------------------------------------------------------------------------
85      -- Step 5: Query the Menu table
```

Results | Chart

	file	status	rows_parsed
1	s3://sfquickstarts/tastybytes/raw_pos/menu/menu.csv.gz	LOADED	100

Query Details

Query duration 1.5s

Rows 1

Query ID 01b2f46b-0001-bb66-0...

```
91  ---> how many rows are in the table?
92  SELECT COUNT(*) AS row_count FROM tasty_bytes_sample_data.raw_pos.menu;
93
94  ---> what do the top 10 rows look like?
95  SELECT TOP 10 * FROM tasty_bytes_sample_data.raw_pos.menu;
96
97  ---> what menu items does the Freezing Point brand sell?
98  SELECT
99     menu_item_name
100 FROM tasty_bytes_sample_data.raw_pos.menu
101 WHERE truck_brand_name = 'Freezing Point';
```

Results | Chart

	ROW_COUNT
1	100

memastikan load berhasil

TEST SNOWFLAKE

```
97   ---> what menu items does the Freezing Point brand sell?
98   SELECT
99      menu_item_name
100  FROM tasty_bytes_sample_data.raw_pos.menu
101  WHERE truck_brand_name = 'Freezing Point';
102
```

Results | Chart

	MENU_ITEM_NAME
1	Lemonade
2	Sugar Cone
3	Waffle Cone
4	Two Scoop Bowl

```
103  ---> what is the profit on Mango Sticky Rice?
104  SELECT
105     menu_item_name,
106     (sale_price_usd - cost_of_goods_usd) AS profit_usd
107  FROM tasty_bytes_sample_data.raw_pos.menu
108  WHERE 1=1
109  AND truck_brand_name = 'Freezing Point'
110  AND menu_item_name = 'Mango Sticky Rice';
111
```

Results | Chart

	MENU_ITEM_NAME	PROFIT_USD
1	Mango Sticky Rice	3.7500

ketika ingin mengetahui list dari hasil data tabel yang ada

```
111
112  ---> to finish, let's extract the Mango Sticky Rice ingredients from the semi-structured column
113  SELECT
114      m.menu_item_name,
115      obj.value:"ingredients"::ARRAY AS ingredients
116  FROM tasty_bytes_sample_data.raw_pos.menu m,
117      LATERAL FLATTEN (input => m.menu_item_health_metrics_obj:menu_item_health_metrics) obj
118  WHERE 1=1
119  AND truck_brand_name = 'Freezing Point'
120  AND menu_item_name = 'Mango Sticky Rice';
```

Results | Chart

	MENU_ITEM_NAME	INGREDIENTS
1	Mango Sticky Rice	["Sweet Mango", "Sticky Salted Rice", "Coconut Milk", "Sesame

Query Details

Query duration 229ms

DigitalSkola
Uncover The World of Digital Skills with Us

THE IMPACT OF SNOWFLAKE AT CAPITAL ONE

Improved productivity and agility

Unlimited Concurrency

New business insights

Fast Elasticity

Performance and access always remain consistent

High Availability

Peace of mind for customers and employees

Comprehensive Security

DigitalSkola
Uncover The World of Digital Skills with Us

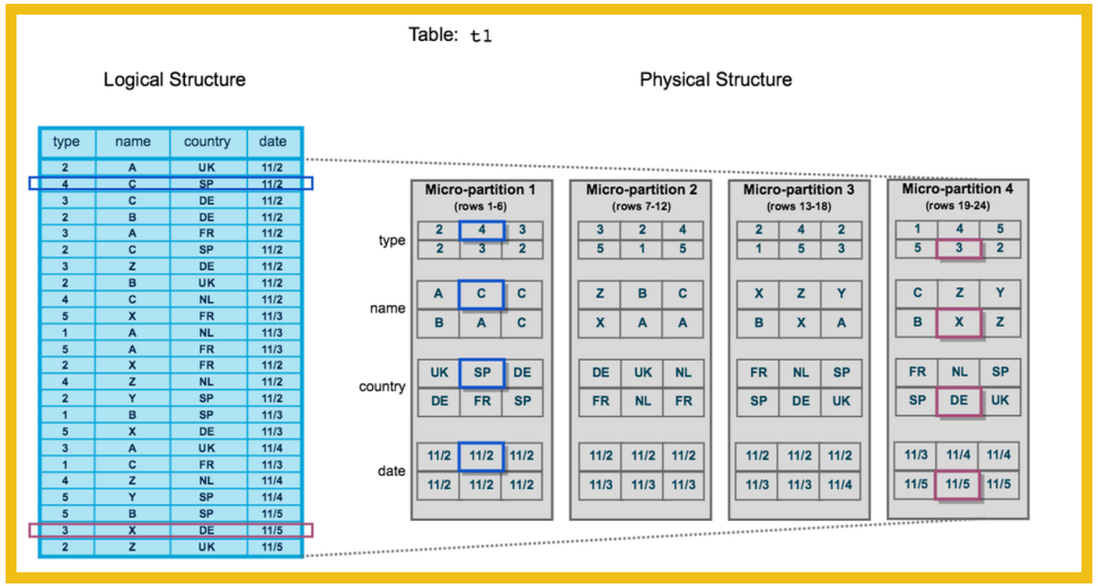
SNOWFLAKE SQL I

ADVANCE SQL (USING SNOWFLAKE)

BEST PRACTICES FOR SNOWFLAKE

1. **Gunakan Auto-Suspend Warehouse**: Hemat biaya dengan menangguhkan warehouse yang tidak aktif secara otomatis.
2. **Kelola Biaya Secara Efektif**: Pahami model penetapan harga Snowflake (penyimpanan vs komputasi) dan gunakan pemantau resource untuk mengoptimalkan penggunaan.
3. **Manfaatkan Profil Kueri Snowflake**: Analisa kueri yang lambat dan identifikasi kemacetan untuk perbaikan.
4. **Transformasi Data Bertahap**: Pisahkan kueri kompleks menjadi potongan-potongan yang lebih kecil dan lebih mudah dikelola untuk kinerja yang lebih baik.
5. **Gunakan Kloning Data**: Buat cadangan instan dengan biaya penyimpanan minimal menggunakan fitur kloning Snowflake.
6. **Manfaatkan Snowpipe**: Aktifkan penyerapan data mendekati real-time dengan Snowpipe untuk meningkatkan ketepatan waktu data.
7. **Implemen Validasi Data**: Pastikan keakuratan dan kelengkapan data dengan pemeriksaan validasi di seluruh proses ETL.
8. **Gunakan Materialized View**: Tingkatkan performa kueri dengan menggunakan materialized view yang telah diperhitungkan sebelumnya.
9. **Implemen Incremental Loading**: Kurangi waktu dan biaya pemrosesan ETL dengan hanya memuat perubahan data sejak pemuatan terakhir.
10. **Gunakan Tabel Eksternal**: Minimalkan biaya penyimpanan dan sederhanakan penyerapan data dengan mereferensikan sumber data eksternal.

DB CLUSTERING



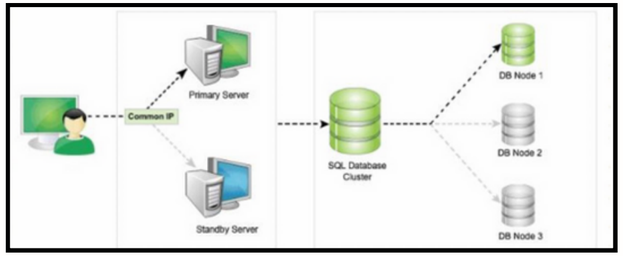
DB Clustering di Snowflake adalah fitur yang memungkinkan untuk mengelompokkan beberapa database (database virtual) bersama-sama untuk meningkatkan kinerja dan skalabilitas. Cluster ini bertindak sebagai satu sistem terpadu, memungkinkan Anda untuk:

- Menjalankan kueri secara paralel di beberapa database
- Meningkatkan throughput dan waktu respons
- Menangani workload yang lebih besar

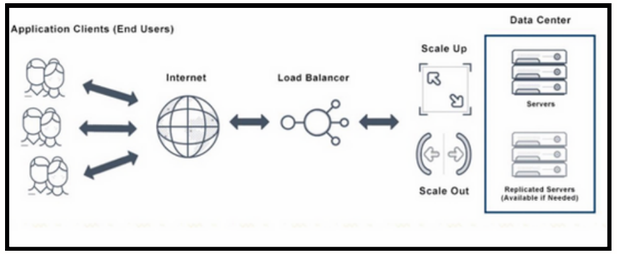
DigitalSkola

TIGA JENIS UTAMA
DB CLUSTERING

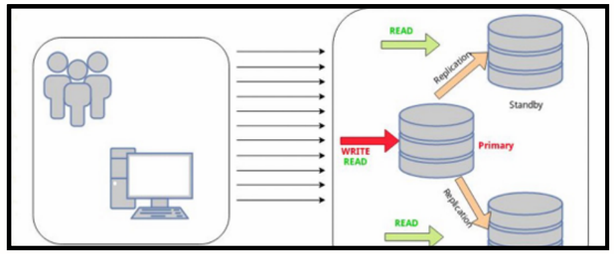
1. **Failover/High Availability Clusters**



2. **High-Performance Clusters**



3. **Load Balancing Clusters**



DigitalSkola

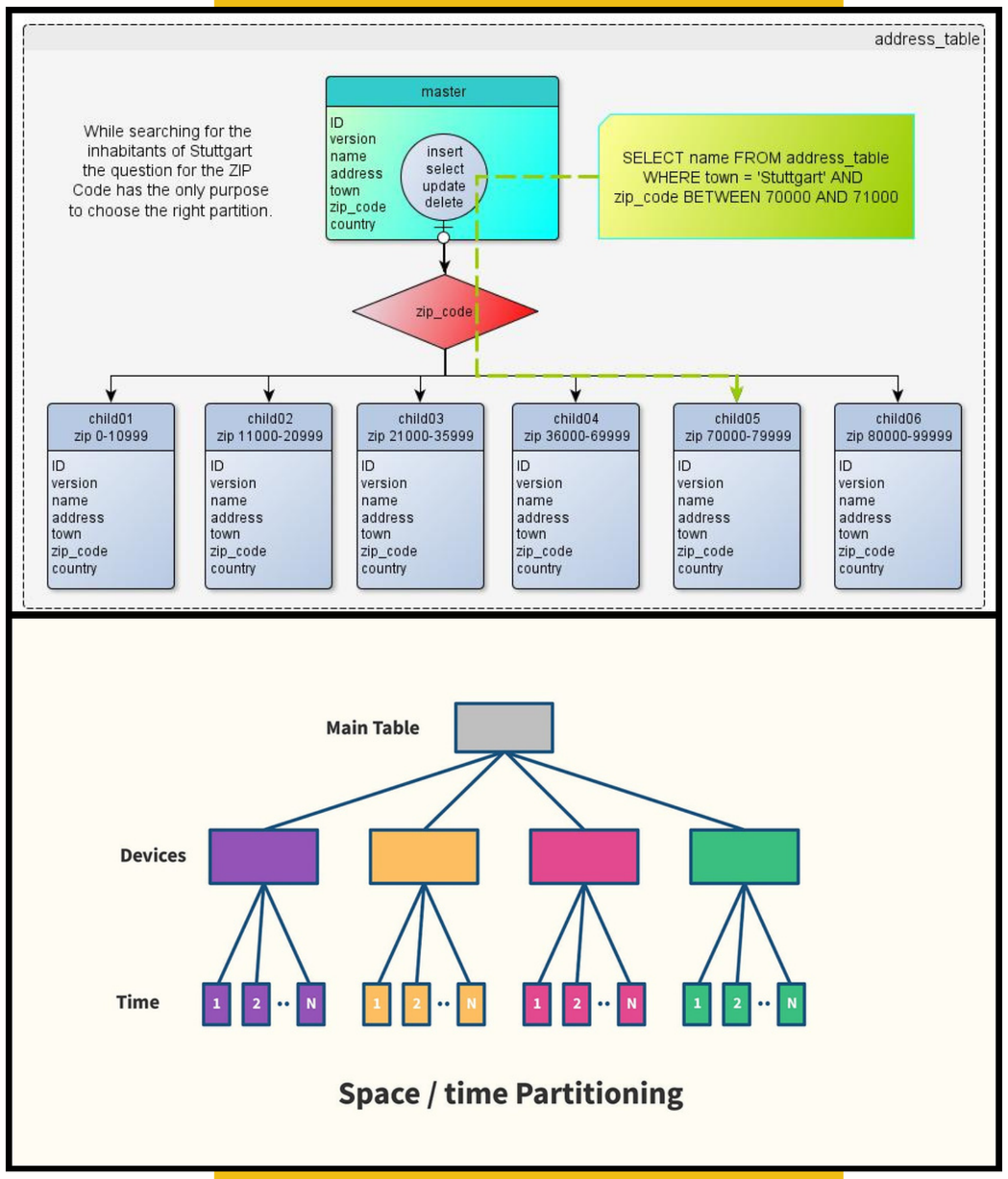


TABLE PARTITION

Table partition adalah teknik membagi tabel data besar menjadi bagian-bagian kecil yang dapat dikelola secara terpisah. Setiap bagian memiliki nama dan karakteristiknya sendiri, memungkinkan manipulasi data secara efisien berdasarkan kriteria tertentu seperti tanggal atau kategori. Ini meningkatkan kinerja, efisiensi, dan skalabilitas sistem basis data.

DigitalSkola

OPERASI JOIN
DI SNOWFLAKE

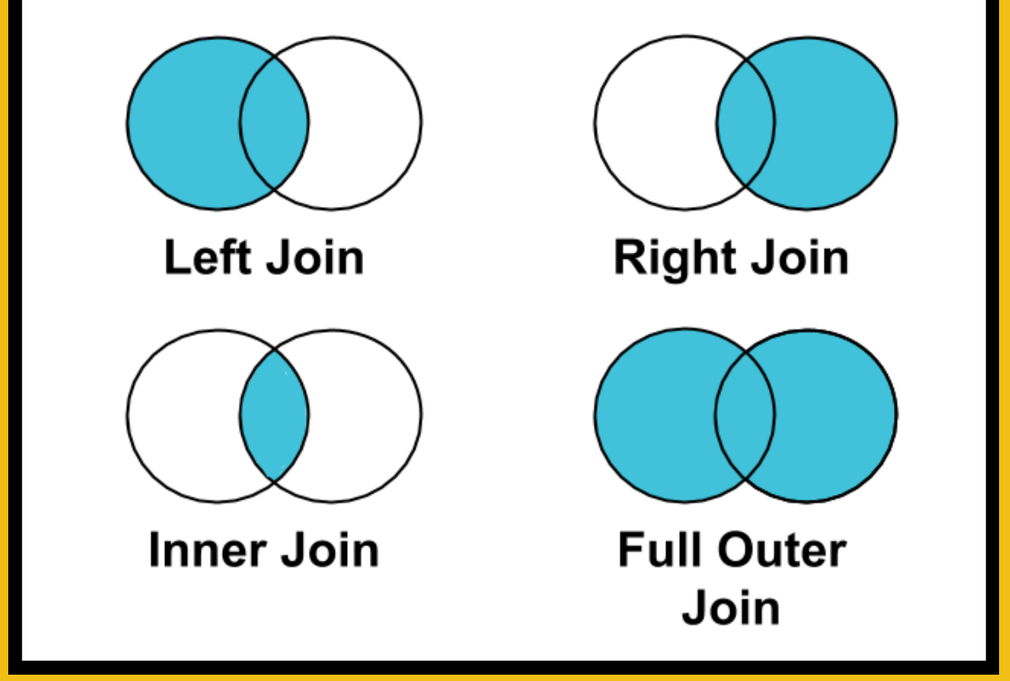
Operasi join adalah salah satu operasi paling penting dalam SQL. Join digunakan untuk menggabungkan data dari dua atau lebih tabel. Ada beberapa jenis join, termasuk:

- **Inner join**: Inner join hanya mengembalikan baris yang ada di kedua tabel.
- **Left join**: Left join mengembalikan semua baris dari tabel kiri, dan baris yang cocok dari tabel kanan.
- **Right join**: Right join mengembalikan semua baris dari tabel kanan, dan baris yang cocok dari tabel kiri.
- **Full join**: Full join mengembalikan semua baris dari kedua tabel, bahkan baris yang tidak cocok di kedua tabel.

```
419  SELECT 'inner join' as method ,count(*) cnt FROM location
420  inner join region_main  rm using(region_id)
421  union all
422  SELECT 'left join no filter' as method , count(*) cnt FROM location
423  left join region_main  rm using(region_id)
424  union all
425  SELECT 'left join filter' as method , count(*) cnt FROM location
426  left join region_main  rm using(region_id)
427  WHERE TRUE
428  and rm.region_id is not null
429  ;
430
```

Objects | Editor | Results | Chart

	METHOD	CNT
1	inner join	20
2	left join no filter	25
3	left join filter	20



SORTING AND GROUPING

PENGURUTAN (SORTING):

- Gunakan klausa ORDER BY untuk mengurutkan hasil akhir kueri Anda.
- Tentukan satu atau lebih kolom setelah ORDER BY untuk mengatur urutan (naik atau turun).

PENGELOMPOKAN (GROUPING):

- Gunakan klausa GROUP BY untuk mengkategorikan baris berdasarkan nilai yang sama dalam satu atau lebih kolom.
- Fungsi agregat seperti COUNT(), SUM(), AVG(), dll. biasanya digunakan dengan pengelompokan untuk meringkas data dalam setiap grup.

SUBQUERY

SUBQUERY

Subquery memang merupakan fitur yang sangat kuat dalam SQL yang memungkinkan untuk melakukan kueri yang lebih kompleks dan dinamis. Mereka dapat digunakan dalam berbagai bagian dari pernyataan SQL dan menawarkan fleksibilitas dalam pengambilan dan manipulasi data

POINT - POINT SUBQUERY

- **Subquery Usage** : Subquery dapat terjadi dalam klausa SELECT, FROM, atau WHERE dari pernyataan SQL.
- **Comparison Operators** : Subquery sering menggunakan operator perbandingan seperti >, <, =, serta operator multi-baris seperti IN, ANY, atau ALL.
- **Execution Order** : query dalam dieksekusi terlebih dahulu sebelum query luar sehingga hasil kueri dalam dapat digunakan oleh query luar.

SUBQUERY

Subquery dalam SELECT clause

```
1  SELECT nama_produk,
2         (SELECT nama_mainan FROM mainan WHERE id = produk.id_mainan) AS mainan
3  FROM produk;
4
5
```

Subquery

Subquery dalam FROM clause

```
1  SELECT AVG(total_harga) AS rata_rata_harga
2  FROM (SELECT SUM(harga) AS total_harga FROM pesanan GROUP BY id_pesanan) AS subquery;
3
```

Subquery dalam WHERE clause

```
1  SELECT nama_pelanggan
2  FROM pelanggan
3  WHERE id_pelanggan IN (SELECT id_pelanggan FROM pesanan WHERE total_harga > 1000);
4
5
```

DigitalSkola

USER DEFINED FUNCTION

User Defined Function (UDF) adalah fungsi yang dibuat oleh pengguna dalam suatu bahasa pemrograman tertentu, seperti SQL, Python, atau JavaScript, untuk menambahkan fungsionalitas khusus yang tidak tersedia dalam fungsi bawaan atau untuk mempermudah penggunaan kode.

Berikut adalah beberapa jenis pada UDF

1 Scalar UDF

Mengembalikan satu nilai tunggal. Misalnya, menghitung total harga dari beberapa item.

2 Table-valued UDF

Mengembalikan tabel sementara yang dapat digunakan dalam pernyataan SQL lainnya. Misalnya, menghasilkan daftar produk berdasarkan kriteria tertentu.

3 Inline Table-valued UDF

Jenis khusus dari table-valued UDF yang berisi satu pernyataan SELECT dan digunakan untuk mengembalikan set hasil langsung.

PENGGUNAAN UDF DENGAN SQL

```sql
CREATE FUNCTION fn_hitung_total_harga (@harga INT, @diskon FLOAT)
RETURNS INT
AS
BEGIN
    DECLARE @total_harga INT;
    SET @total_harga = @harga - (@harga * @diskon);
    RETURN @total_harga;
END;

```

fungsi dipanggil dengan query

```sql
SELECT nama_produk, harga, dbo.fn_hitung_total_harga(harga, 0.1) AS
total_setelah_diskon
FROM produk;

```

DigitalSkola

PENGGUNAAN UDF DENGAN PYTHON

fungsi dipanggil dengan query

```python
# Membuat UDF untuk menghitung luas persegi
def hitung_luas_persegi(sisi):
    luas = sisi * sisi
    return luas
```

```python
# Memanggil UDF untuk menghitung luas persegi dengan sisi 5
luas_persegi = hitung_luas_persegi(5)
print("Luas persegi dengan sisi 5 adalah:", luas_persegi)
```

DigitalSkola

DigitalSkola
Uncover The World of Digital Skills with Us

SNOWFLAKE
SQL II

TASTY BYTES

Tasty Bytes adalah merek food truck fiktif yang dibuat oleh tim frostbyte di Snowflake.

Tasty bytes digunakan dalam berbagai tutorial dan quickstart untuk memperkenalkan dan menunjukkan penggunaan berbagai fitur dan fungsi dalam Snowflake.

Fitur dan fungsi yang diperkenalkan

- FINANCIAL GOVERNANCE
- TRANSFORMATION
- SEMI-STRUCTURED DATA
- DATA GOVERNANCE
- COLLABORATION
- GEOSPATIAL

FINANCIAL GOVERNANCE

Financial governance adalah salah satu aspek yang penting dalam manajemen dan pengawasan sumber daya komputasi dan keuangan dalam platform snowflake.

Berikut adalah beberapa hal yang dipelajari pada bagian financial governance

1

Snowflake Virtual Warehouse

2

Resource Monitors

3

Account and Warehouse Level Timeout Parameters.

FINANCIAL GOVERNANCE

Create Warehouse

```
-- Section 3: Step 1 - Role and Warehouse Context
USE ROLE tasty_admin;
USE WAREHOUSE tasty_de_wh;


-- Section 3: Step 2 - Creating and Configuring a Warehouse
CREATE OR REPLACE WAREHOUSE tasty_test_wh WITH
COMMENT = 'test warehouse for tasty bytes'
    WAREHOUSE_TYPE = 'standard'
    WAREHOUSE_SIZE = 'xsmall'
    MIN_CLUSTER_COUNT = 1
    MAX_CLUSTER_COUNT = 2
    SCALING_POLICY = 'standard'
    AUTO_SUSPEND = 60
    AUTO_RESUME = true
    INITIALLY_SUSPENDED = true;
```

Create Resource Monitor

```
-- Section 4: Step 1 - Creating a Resource Monitor
USE ROLE accountadmin;
CREATE OR REPLACE RESOURCE MONITOR tasty_test_rm
WITH
    CREDIT_QUOTA = 100 -- 100 credits
    FREQUENCY = monthly -- reset the monitor monthly
    START_TIMESTAMP = immediately -- begin tracking immediately
    TRIGGERS
        ON 75 PERCENT DO NOTIFY -- notify accountadmins at 75%
        ON 100 PERCENT DO SUSPEND -- suspend warehouse at 100 percent, let queries finish
        ON 110 PERCENT DO SUSPEND_IMMEDIATE; -- suspend warehouse and cancel all queries at 110 percent


-- Section 4: Step 2 - Applying our Resource Monitor to our Warehouse
ALTER WAREHOUSE tasty_test_wh SET RESOURCE_MONITOR = tasty_test_rm;
```

FINANCIAL GOVERNANCE

```
-- Section 5: Step 1 - Exploring Warehouse Statement Parameters
SHOW PARAMETERS LIKE '%statement%' IN WAREHOUSE tasty_test_wh;


-- Section 5: Step 2 - Adjusting Warehouse Statement Timeout Parameter
ALTER WAREHOUSE tasty_test_wh SET statement_timeout_in_seconds = 1800;


-- Section 5: Step 3 - Adjusting Warehouse Statement Queued Timeout Parameter
ALTER WAREHOUSE tasty_test_wh SET statement_queued_timeout_in_seconds = 600;
```

Protecting Warehouse from Long Running Queries

Protecting Account from Long Running Queries

```
-- Section 6: Step 1 - Adjusting the Account Statement Timeout Parameter
ALTER ACCOUNT SET statement_timeout_in_seconds = 18000;


-- Section 6: Step 2 - Adjusting the Account Statement Queued Timeout Parameter
ALTER ACCOUNT SET statement_queued_timeout_in_seconds = 3600;
```

TRANSFORMATION

Transformation adalah adalah proses di mana data yang telah diambil pada proses ekstraksi akan diolah dan diubah dari bentuk asli menjadi bentuk yang sesuai dengan kebutuhan data warehouse.

Zero-Copy Cloning

Fitur ini memungkinkan pengguna untuk membuat *copy* dari berbagai tabel, skema atau seluruh database tanpa adanya tambahan biaya karena masih menggunakan storage yang sama untuk berbagi penyimpanan.

Time Travel

Fitur ini memungkinkan akses ke data historis yang dapat mengembalikan objek data yang mungkin telah dihapus secara tidak sengaja atau disengaja, menduplikasi data dan menganalisis penggunaan data selama periode waktu tertentu.

TRANSFORMATION

```
-- Section 3: Step 1 - Create a Clone of Production
USE ROLE tasty_dev;

CREATE OR REPLACE TABLE frostbyte_tasty_bytes.raw_pos.truck_dev
    CLONE frostbyte_tasty_bytes.raw_pos.truck;
```

Zero-copy cloning

```
-- Section 4: Step 1 - Querying our Cloned Table
USE WAREHOUSE tasty_dev_wh;

SELECT
    t.truck_id,
    t.year,
    t.make,
    t.model
FROM frostbyte_tasty_bytes.raw_pos.truck_dev t
ORDER BY t.truck_id;
```

```
-- Section 4: Step 2 - Re-Running our Query
SELECT
    t.truck_id,
    t.year,
    t.make,
    t.model
FROM frostbyte_tasty_bytes.raw_pos.truck_dev t
ORDER BY t.truck_id;
```

DigitalSkola

TRANSFORMATION

Time Travel for Data Disaster Recovery

```sql
-- Section 7: Step 1 - Leveraging Query History
SELECT
    query_id,
    query_text,
    user_name,
    query_type,
    start_time
FROM TABLE(frostbyte_tasty_bytes.information_schema.query_history())
WHERE 1=1
    AND query_type = 'UPDATE'
    AND query_text LIKE '%frostbyte_tasty_bytes.raw_pos.truck_dev%'
ORDER BY start_time DESC;


-- Section 7: Step 2 - Setting a SQL Variable
SET query_id =
(
    SELECT TOP 1 query_id
    FROM TABLE(frostbyte_tasty_bytes.information_schema.query_history())
    WHERE 1=1
        AND query_type = 'UPDATE'
        AND query_text LIKE '%SET truck_age = (YEAR(CURRENT_DATE()) / t.year);'
    ORDER BY start_time DESC
);


-- Section 7: Step 3 - Leveraging Time-Travel to Revert our Table
CREATE OR REPLACE TABLE frostbyte_tasty_bytes.raw_pos.truck_dev
    AS
SELECT * FROM frostbyte_tasty_bytes.raw_pos.truck_dev
BEFORE(STATEMENT => $query_id);
```

```sql
-- Section 8: Step 1 - Adding Correctly Calculated Values to our Column
UPDATE frostbyte_tasty_bytes.raw_pos.truck_dev t
SET truck_age = (YEAR(CURRENT_DATE()) - t.year);


-- Section 8: Step 2 - Swapping our Development Table with Production
USE ROLE sysadmin;

ALTER TABLE frostbyte_tasty_bytes.raw_pos.truck_dev
    SWAP WITH frostbyte_tasty_bytes.raw_pos.truck;


-- Section 8: Step 3 - Validate Production
SELECT
    t.truck_id,
    t.year,
    t.truck_age
FROM frostbyte_tasty_bytes.raw_pos.truck t
WHERE t.make = 'Ford';
```

SEMI-STRUCTURED DATA

Semi-Structured Data merupakan data yang tidak memiliki struktur tabel relasional tetapi memiliki beberapa aspek organisasi yang mencakup tag, tanda, atau pengenal lainnya. Contoh tipe data semi-structured ini adalah JSON, Avro, dan XML.

Beberapa hal yang dipelajari mengenai semi-structured data dalam snowflake

- Snowflake VARIANT Data Type
- Semi-Structured Data Processing via Dot notation
- Lateral Flattening

SEMI-STRUCTURED DATA

Profiling Menu Data

```
-- Section 3: Step 1 - Setting our Context and Querying our Table
USE ROLE tasty_data_engineer;
USE WAREHOUSE tasty_de_wh;

SELECT TOP 10
    m.truck_brand_name,
    m.menu_type,
    m.menu_item_name,
    m.menu_item_health_metrics_obj
FROM frostbyte_tasty_bytes.raw_pos.menu m;


-- Section 3: Step 2 - Exploring our Semi-Structured Column
SHOW COLUMNS IN frostbyte_tasty_bytes.raw_pos.menu;


-- Section 3: Step 3 - Traversing Semi-Structured Data using Dot Notation
SELECT
    m.menu_item_health_metrics_obj:menu_item_id AS menu_item_id,
    m.menu_item_health_metrics_obj:menu_item_health_metrics AS menu_item_health_metrics
FROM frostbyte_tasty_bytes.raw_pos.menu m;
```

DigitalSkola

SEMI-STRUCTURED DATA

Flattening Semi-Structured Data

```
SELECT
    m.menu_item_name,
    obj.value:"ingredients"::VARIANT AS ingredients
FROM frostbyte_tasty_bytes.raw_pos.menu m,
    LATERAL FLATTEN (input => m.menu_item_health_metrics_obj:menu_item_health_metrics) obj;


-- Section 4: Step 2 - Exploring an Array Function
SELECT
    m.menu_item_name,
    obj.value:"ingredients"::VARIANT AS ingredients
FROM frostbyte_tasty_bytes.raw_pos.menu m,
    LATERAL FLATTEN (input => m.menu_item_health_metrics_obj:menu_item_health_metrics) obj
WHERE ARRAY_CONTAINS('Lettuce'::VARIANT, obj.value:"ingredients"::VARIANT);


-- Section 4: Step 3 - Structuring Semi-Structured Data at Scale
SELECT
    m.menu_item_health_metrics_obj:menu_item_id::integer AS menu_item_id,
    m.menu_item_name,
    obj.value:"ingredients"::VARIANT AS ingredients,
    obj.value:"is_healthy_flag"::VARCHAR(1) AS is_healthy_flag,
    obj.value:"is_gluten_free_flag"::VARCHAR(1) AS is_gluten_free_flag,
    obj.value:"is_dairy_free_flag"::VARCHAR(1) AS is_dairy_free_flag,
    obj.value:"is_nut_free_flag"::VARCHAR(1) AS is_nut_free_flag
FROM frostbyte_tasty_bytes.raw_pos.menu m,
    LATERAL FLATTEN (input => m.menu_item_health_metrics_obj:menu_item_health_metrics) obj;
```

SEMI-STRUCTURED DATA

Creating Structured Views Over Semi-Structured Data

```
-- Section 5: Step 1 - Creating our Harmonized View Using our Semi-Structured Flattening SQL
CREATE OR REPLACE VIEW frostbyte_tasty_bytes.harmonized.menu_v
    AS
SELECT
    m.menu_id,
    m.menu_type_id,
    m.menu_type,
    m.truck_brand_name,
    m.menu_item_health_metrics_obj:menu_item_id::integer AS menu_item_id,
    m.menu_item_name,
    m.item_category,
    m.item_subcategory,
    m.cost_of_goods_usd,
    m.sale_price_usd,
    obj.value:"ingredients"::VARIANT AS ingredients,
    obj.value:"is_healthy_flag"::VARCHAR(1) AS is_healthy_flag,
    obj.value:"is_gluten_free_flag"::VARCHAR(1) AS is_gluten_free_flag,
    obj.value:"is_dairy_free_flag"::VARCHAR(1) AS is_dairy_free_flag,
    obj.value:"is_nut_free_flag"::VARCHAR(1) AS is_nut_free_flag
FROM frostbyte_tasty_bytes.raw_pos.menu m,
    LATERAL FLATTEN (input => m.menu_item_health_metrics_obj:menu_item_health_metrics) obj;


-- Section 5: Step 2 - Promoting from Harmonized to Analytics with Ease
CREATE OR REPLACE VIEW frostbyte_tasty_bytes.analytics.menu_v
COMMENT = 'Menu level metrics including Truck Brands and Menu Item details including cost, price, ingredients and dietary restrictions'
    AS
SELECT
    *
    EXCLUDE (menu_type_id) --exclude MENU_TYPE_ID
    RENAME  (truck_brand_name AS brand_name) -- rename TRUCK_BRAND_NAME to BRAND_NAME
FROM frostbyte_tasty_bytes.harmonized.menu_v;
```

TERIMA KASIH

DigitalSkola
Uncover The World of Digital Skills with Us