LEARNING PROGRESS REVIEW

--Week 2--

Kelompok 7 - Citizen Data Scientist

Diaz Jubiary - Hermulia Hadie
Desi Sulistyowati - Farahul Jannah

Daftar Isi

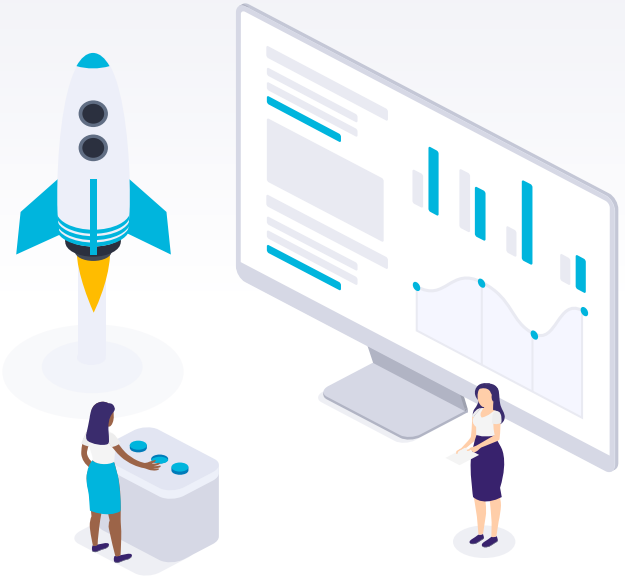
Key Presentation Point

- Introduction to Data & Database
  - Data & Database
  - DBMS
  - Tipe Data and ERD
- Basic SQL
  - Pengertian SQL
  - Schema and Table
  - DDL Statements (SELECT, ALTER, DROP, etc)
  - Data Lifecycle
- Intermediate SQL
  - Aggregate Function (SUM, MIN, AVG etc)
  - String Function (“||”, LOWER, POSTION, etc)
  - Grouping and Filtering
  - Query Process Order

DigitalSkola

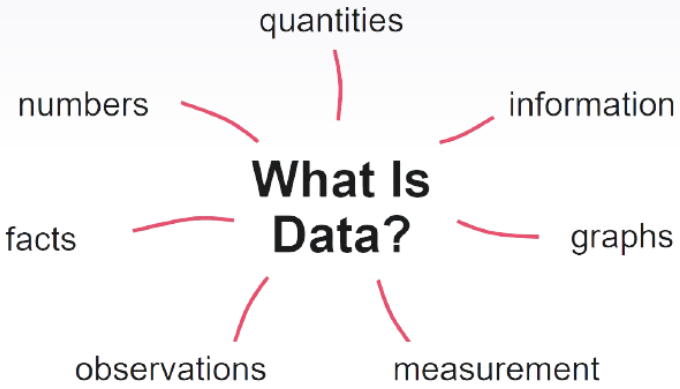
1

Introduction to Data & Database



Apa itu Data?

- Data berasal dari kata jamak “datum” yang berasal dari Bahasa Latin yang artinya “sesuatu yang diberikan”.
- Data merupakan fakta yang dapat direkam dan disimpan pada media computer.
- Dalam kehidupan sehari-hari, kita menggunakan kata data untuk menggambarkan fakta tentang orang, peristiwa, tempat, atau benda apa pun. Data adalah fakta mentah (raw facts) yang mungkin berupa angka, nilai, nama, tanggal, dll.



DigitalSkola

Apa itu Database?

- Tempat penyimpanan dan memproses data elektornik yang berisi satu atau banyak file .
- Database berisi data yang akurat, konsisten dan non-redundant (tidak berlebihan).
- Data dalam database digunakan oleh program aplikasi yang berbeda.

Database Management System (DBMS)

- Database adalah suatu program yang bekerja sama dengan sistem operasi untuk membuat, memproses, menyimpan, mengambil, mengontrol, dan mengelola data.

Fungsi DBMS :

- Mendefinisikan, membuat dan mengatur database: DBMS menetapkan hubungan logis antara elemen data yang berbeda dalam database dan juga mendefinisikan skema dan subskema menggunakan DDL.
- Input data : DBMS berfungsi memasukkan data ke dalam database melalui perangkat input dengan bantuan user.
- Proses data : berfungsi memanipulasi dan pemrosesan data yang disimpan dalam database menggunakan DML.
- Menjaga integritas dan keamanan data : hanya memberikan akses kepada pengguna yang berwenang saja ke dalam database sehingga data terjaga integritas dan keamanannya.

Komponen DBMS

Structured Query Language (SQL)

bahasa yang digunakan untuk mengakses data dalam suatu database

Data Definition Language (DDL)

mendefiniskan database, menentukan tipe data, struktur data dan batasan pada data yang akan disimpan dalam database.misalnya :
CREATE, RENAME, ALTER, DROP

Data Manipulation Language (DML)

perintah SQL yang berhubungan dengan manipulasi atau pengolahan data dalam table misalnya :
SELECT, INSERT, UPDATE, DELETE

Jenis Data

Bilangan Bulat (Integer)

- Tipe data numerik yang digunakan pada bilangan bulat atau bilangan tanpa desimal.
- Misalnya 7, 22, 23, 27, 1997
- Bilangan ini mengenal nilai positif dan negatif

Tipe bilangan bulat (integer) :

Tipe Data	Ukuran (bit)	Range
byte	8	-128 hingga 127
short	16	-32768 hingga 32767
int	32	-2147483648 hingga 2147483647
long	64	-9223372036854775808 hingga 9223372036854775807

Jenis Data

Floating Point (Bilangan Pecahan)

- Tipe data numerik yang digunakan pada bilangan pecahan atau bilangan desimal
- Misalnya 3,14; 9,7; 15,78
- Tipe bilangan pecahan (floating point) :

Tipe Data	Ukuran (bit)	Range	Jumlah Digit
Float	32	-3,4 x 1038 hingga 3,4 x 1038	6 hingga 7 digit
Double	64	-1,8 x 10308 hingga 1,8 x 10308	15 digit

Jenis Data

- Karakter (Char)
  - Tipe data yang memungkinkan untuk menyimpan informasi berupa karakter.
  - Karakter tersebut bisa berupa huruf, angka, tanda baca dan karakter spesial.
  - Biasanya didefinisikan dengan tanda petik (') di awal dan akhir.

- String

Jika char hanya bisa merepresentasikan satu karakter saja, string dapat digunakan untuk menyimpan sekumpulan karakter.

Char vs String

- Char = A, B, C, D, E, 1, 7, 4
- String = 'Char' , 'Data', 'Jenis Data'

- Boolean

tipe yang memiliki dua nilai yaitu benar (true) atau salah (false).

Entity-Relationship

- Model Entity-Relationship (ER) digunakan untuk komunikasi antara database designer dengan end user pada saat proses pengembangan database.
- Konstruksi ER, yaitu entitas (entity), hubungan (relationship), dan atribut (attribute).

Entitas (Entity)

Data utama yang berisi data mana yang dikumpulkan, misalnya, orang, tempat, benda, atau peristiwa.

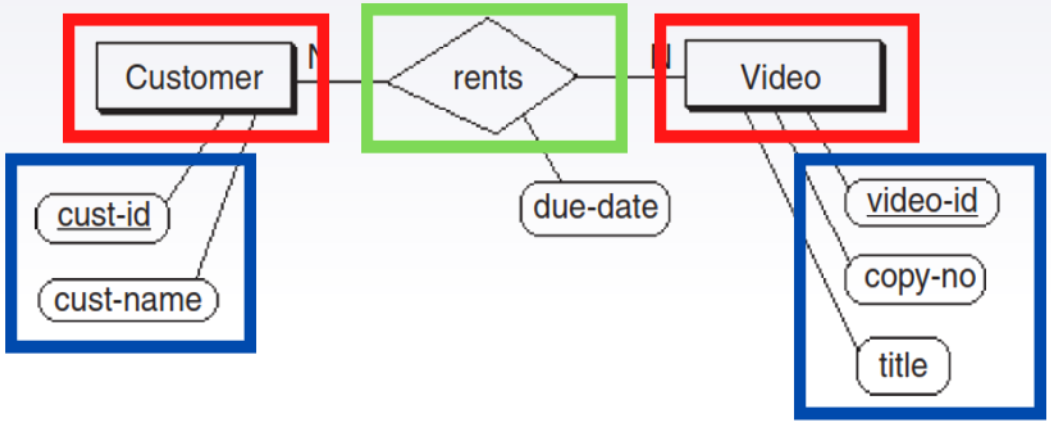
Hubungan (Relationship)

hubungan antar satu atau lebih entitas (entity) .

Atribut (Attribute)

informasi detail tentang suatu entitas.

Entity-Relationship



Keterangan

: Entitas (entity)

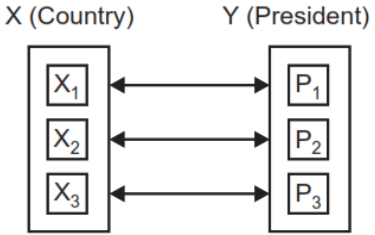
: Hubungan (relationship)

: Antribut (attribute)

Cardinality Relationship

One to One (1 : 1)

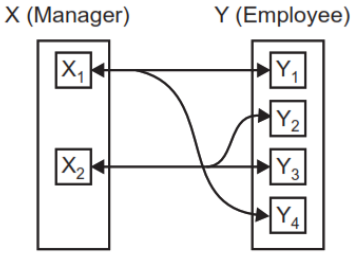
- Satu entitas di X berhubungan dengan maksimal satu entitas di Y
- Satu entitas di Y berhubungan dengan maksimal satu entitas di X



satu negara hanya mempunyai sati presiden

One to Many (1 : M)

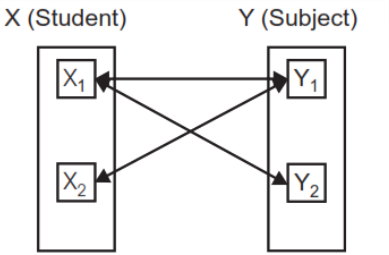
- Satu entitas di X berhubungan dengan sejumlah entitas di Y
- Satu entitas di Y berhubungan dengan maksimal satu entitas di X



satu manajer mempunyai beberapa karyawan tapi satu karyawan hanya bekerja di bawah satu manajer

Many to Many (M : M)

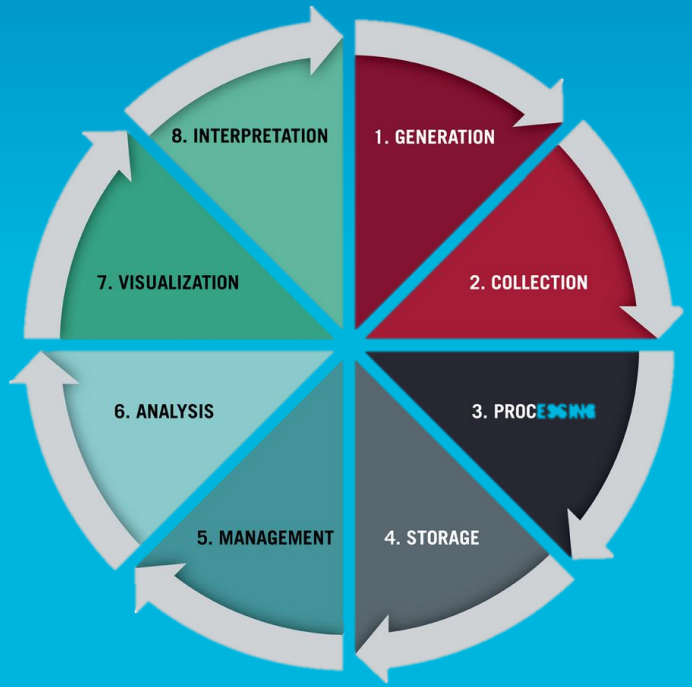
- Satu entitas di X berhubungan dengan beberapa (nol atau lebih) entitas di Y



satu siswa dapat mengikuti beberapa mata pelajaran dan satu mata pelajaran dapat diikuti oleh beberapa siswa

DigitalSkola

Data Lifecycle



- Generation
- Collection
- Processing
- Storage
- Management
- Analysis
- Visualization
- Interpretation

Data Lifecycle

Generation

Proses pembuatan data, biasanya data dihasilkan oleh proses penjualan, pembelian, perekrutan, komunikasi ataupun interaksi lainnya

Collection

Proses pemilihan data apa saja yang dibutuhkan dari berbagai macam data yang tersedia. Teknik pengumpulan data: formulir,survey, wawancara, pengamatan langsung

Processing

Beberapa kegiatan yang dilakukan pada tahap processing : data wraggling (data cleaning), data compression, data encryption

Storage

Kumpulan data disimpan dalam database, dapat disimpan dalam benntuk cloud, server, atau hard drive

Management

Melibatkan pengorganisasian, penyimpanan, dan pengambilan data yang diperlukan selama umur proyek data

Analysis

Mengumpulkan insight dari raw data, metode yang umum digunakan yaitu pemodelan statistik, algoritma, kecerdasan buatan, penambangan data (data mining), & pembelajaran mesin (machine learning)

Visualization

Pembuatan representasi grafis dari informasi yang telah ditemukan, bertujuan untuk memudahkan dalam mengkomunikasikan informasi yang didapat ke audiens

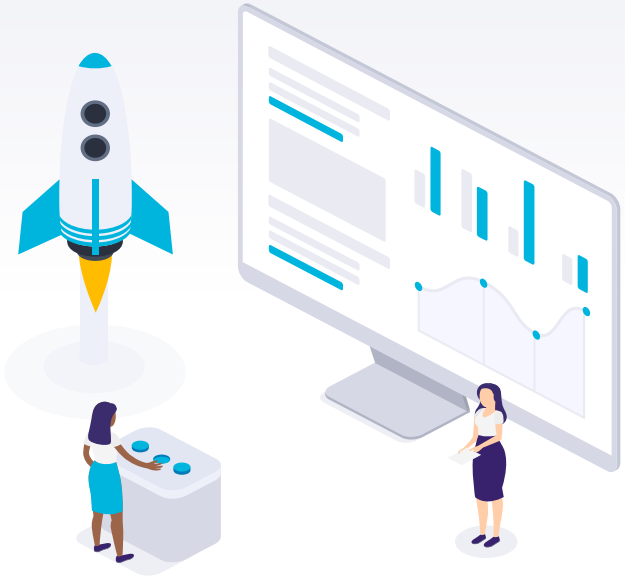
Interpretation

Memberikan pemahaman akan analisis dan visualisasi yang telah dilakukan sebelumnya. Pada tahap ini sudah ditemukan jawaban atas permasalahan yang sedang dihadapi

DigitalSkola

2

Basic SQL



Apa itu SQL

- SQL adalah kepanjangan dari Structured Query Languange.
- SQL memungkinkan kita mengakses dan memanipulasi database.
- Penggunaan bahasa SQL sangat mendekati penggunaan bahasa manusia sehinga relatif lebih muda dipelajari dibanding bahasa lain.
- SQL menjadi standar American National Standards Institute (ANSI) pada 1986, dan International Organization for Standardization (ISO) pada 1987, hingga sekarang masih banyak digunakan di berbagai industri dan bidang.
- Meskipun SQL adalah standar ANSI/ISO, ada beberapa versi bahasa SQL yang berbeda seperti SQL, T-SQL, PL-SQL. Pada beberapa tools dbm, terdapat beberapa perbedaan syntax penulisan.

Kenapa SQL itu Penting ?

- SQL merupakan bahasa terpopuler kedua setelah python di dalam bidang Data Science.
- SQL merupakan *scripting languange* yang sangat populer di dunia korporasi karena kegunaannya untuk berkomunikasi dengan database.

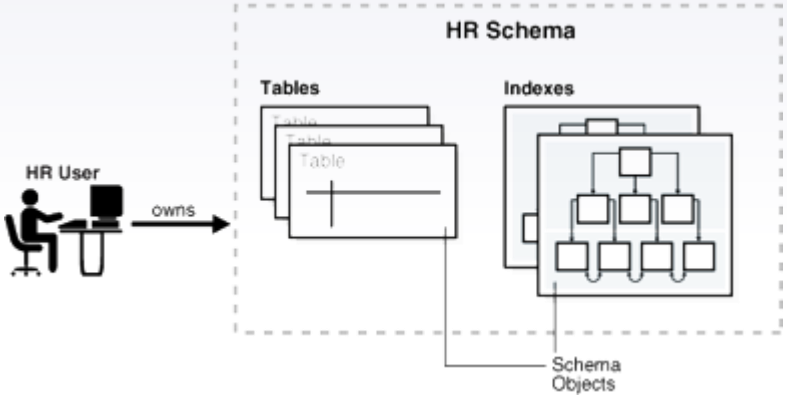
Hampir semua data kita disimpan di database seperti :

1. Informasi identitas pribadi)
2. Perilaku di situs web
3. Riwayat transaksi di e-comms
4. Laporan bank
5. Obrolan, postingan, dan aktivitas di media sosial
6. Foto, dokumen, lagu kita di cloud

Ada banyak tools DBMS yang dapat digunakan

SCHEMA

Schema database adalah *logical container* untuk struktur data, yang disebut objek skema. Contoh objek skema adalah tabel dan indeks. Objek skema dibuat dan dimanipulasi dengan SQL.



Terletak di bawah Database dan biasanya dibuat untuk tujuan tertentu. Misalnya, database perusahaan dapat mencakup skema untuk keuangan, HR, alat pihak ketiga, data yang dibersihkan, gudang data, dll.

SCHEMA SYNTAX

- Putuskan apakah kita ingin menggunakan camelCase atau snake_case.
- Hanya gunakan huruf dan angka.
- Gunakan garis bawah jika kita menggunakan snake_case.
- Gunakan nama kolom yang sederhana dan deskriptif.
- Satu sumber kebenaran! Jangan membuat banyak versi.

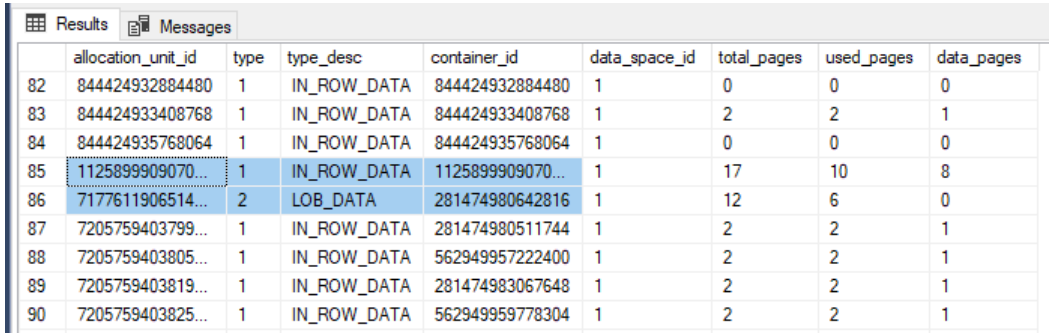
```
CREATE SCHEMA IF NOT EXISTS schema_name
```



```
CREATE SCHEMA IF NOT EXISTS digital_skola
```

TABLE

- Unit dasar penyimpanan data di Database. Data dalam tabel disimpan dalam baris dan kolom seperti spreadsheet.
- Terletak di bawah Skema dan dibuat untuk menyimpan data dalam format tabel. Misalnya, skema Penjualan dapat mencakup tabel transaksi, tabel produk, tabel toko, dll.



Results | Messages

	allocation_unit_id	type	type_desc	container_id	data_space_id	total_pages	used_pages	data_pages
82	844424932884480	1	IN_ROW_DATA	844424932884480	1	0	0	0
83	844424933408768	1	IN_ROW_DATA	844424933408768	1	2	2	1
84	844424935768064	1	IN_ROW_DATA	844424935768064	1	0	0	0
85	1125899909070...	1	IN_ROW_DATA	1125899909070...	1	17	10	8
86	7177611906514...	2	LOB_DATA	281474980642816	1	12	6	0
87	7205759403799...	1	IN_ROW_DATA	281474980511744	1	2	2	1
88	7205759403805...	1	IN_ROW_DATA	562949957222400	1	2	2	1
89	7205759403819...	1	IN_ROW_DATA	281474983067648	1	2	2	1
90	7205759403825...	1	IN_ROW_DATA	562949959778304	1	2	2	1

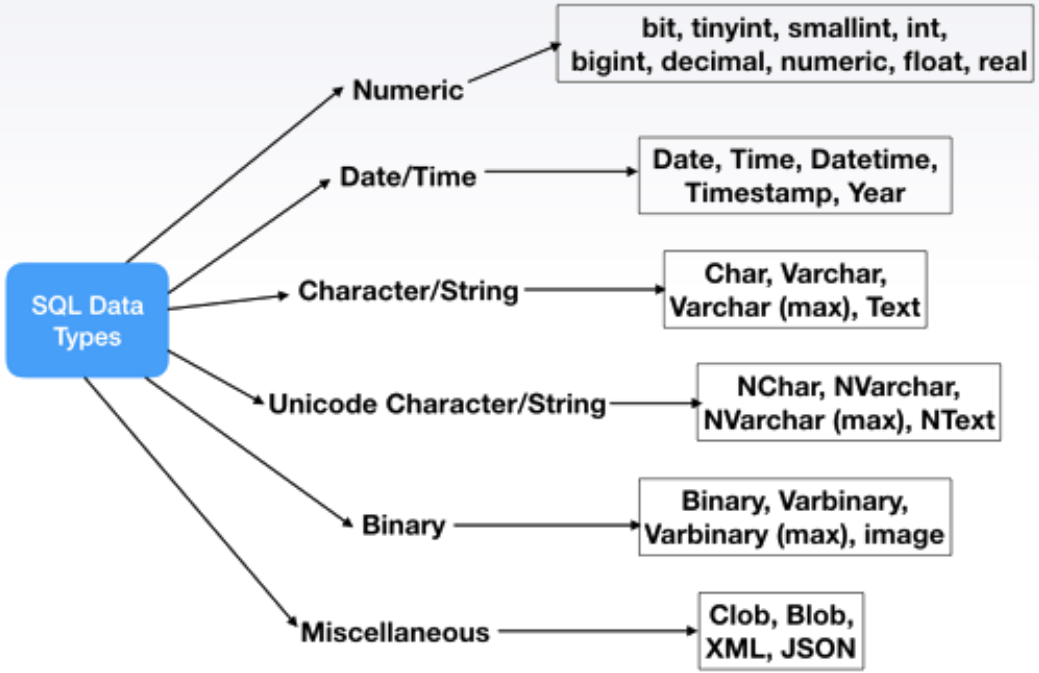
TABLE SYNTAX

- Gunakan tips yang sama seperti membuat skema.
- Pilih tipe data yang tepat dan berikan panjang karakter yang diperlukan untuk setiap kolom.
- Pilih batasan yang diperlukan (NOT NULL, UNIQUE, PRIMARY KEY).

```
CREATE TABLE [IF NOT EXISTS] table_name (
  column1 datatype(length) column_constraint,
  column2 datatype(length) column_constraint,
  column3 datatype(length) column_constraint,
  -- PRIMARY KEY (column1)
  -- FOREIGN KEY (column2)
       REFERENCES table_name(column_name)
)
```

```
create table if not exists batch_11.homework_2_group_7
(
id int primary key not null,
nama_depan varchar(255) not null,
nama_belakang varchar(255) not null,
alamat_email varchar(255) not null,
link_linkedIn varchar(255) not null,
pekerjaan_impian varchar(255) not null,
alasan_mengikuti_bootcamp varchar(255) not null
)
```

DATABASE – Data Types



IMPORT DATA

Untuk menarik data dari direktori lokal ke database kita dapat membuat tabel baru atau menyisipkan ke tabel yang ada secara manual atau menggunakan alat yang disediakan di dalam aplikasi DBMS

1. Pastikan semua format sudah sesuai terutama tanggal dan/atau waktu.
2. Jika kita ingin menyisipkan data ke tabel yang ada, pastikan kolom sudah cocok.

INSERT INTO

- Digunakan untuk mengisi data secara manual atau menambahkan data dari tabel lain.
- Pastikan jumlah kolom cocok antara sumber dan target, dan selalu meletakkan nama skema di depan nama tabel.

```
INSERT INTO table_name VALUES (data1, data2,....),
(data1, data2,....)

INSERT INTO table_name (column1, column2)
VALUES(data1, data2,....), (data1, data2,....)

INSERT INTO table_name
SELECT * FROM table_name

INSERT INTO table_name (column1, column2,....)
SELECT column1, column2,.... FROM table_name
```

```
insert into batch_11.anggota_diaz_jubairy values
(3,'Diaz Jubairy', '082213445022', current_date)
```

DigitalSkola

SELECT

- Digunakan untuk melihat tabel dalam kondisi tertentu.
- Lebih baik menulisakan kolom daripada menggunakan (*)

```
SELECT * FROM table_name;

SELECT column1, column2, ....
FROM table_name;
```

```
select * from batch_11.anggota_diaz_jubairy

select first_name, last_name, hobby
from batch_11.anggota_diaz_jubairy
```

ALTER

- Digunakan untuk mengubah beberapa perubahan ke dalam skema. Kita bisa menambah dan/atau menghapus kolom menggunakan fungsi ini.
- Pilih tipe data yang tepat dan berikan panjang karakter yang diperlukan untuk setiap kolom.
- Pastikan untuk mengisi nama kolom yang benar karena kita tidak dapat mengulang perubahan.

```
ALTER TABLE table_name ADD column datatype(length);

ALTER TABLE table_name DROP COLUMN column1;
```

```
alter table batch_11.employee_diaz_jubairy add
hobby varchar(255)

alter table batch_11.employee_diaz_jubairy
drop column commission_pct,
drop column manager_id,
drop column department_id;
```

DELETE

- Digunakan untuk menghapus baris dalam tabel dalam kondisi tertentu.
- Pastikan untuk mengisi nama kolom yang benar karena kita tidak dapat mengulang perubahan.

```
DELETE FROM table_name
WHERE condition;
```

```
delete from batch_11.employee_diaz_jubairy
where employee_id != 100 and first_name != 'Diaz'
```

TRUNCATE

- Digunakan untuk menghapus tabel dalam kondisi tertentu.
- Pastikan untuk mengisi nama kolom yang benar karena kita tidak dapat mengulang perubahan.

```
TRUNCATE TABLE table_name;
```

```
truncate table batch_11.employee_diaz_jubairy
```

DROP

- Digunakan untuk menghapus tabel tanpa syarat.
- Pastikan untuk mengisi nama kolom yang benar karena kita tidak dapat mengulang perubahan

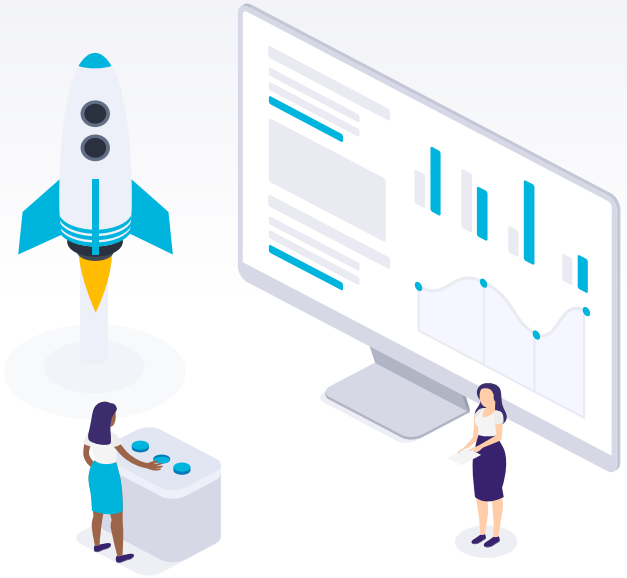
```
DROP TABLE table_name
```

```
drop table batch_11.employee_diaz_jubairy
```

DigitalSkola

3

Intermediate SQL



General Function

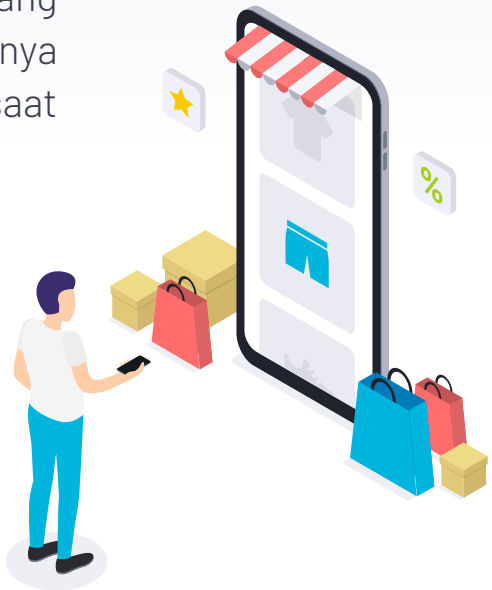
Penulisan fungsi-fungsi dalam SQL harus berdasarkan format-format tertentu dan urutan penulisan sesuai aturan SQL. Kesalahan penulisan format atau urutan penulisan script akan menyebabkan error ketika di run. Beberapa perintah umum dalam SQL adalah sebagai berikut:

```
SELECT column1, column2, ....
FROM table_name
WHERE condition (s)
GROUP BY field_name (s)
HAVING condition (s)
ORDER BY field_name (s)
LIMIT number
```

Distinct

Perintah distinct digunakan untuk menghilangkan data duplikat pada suatu kolom. Pada suatu kolom (field), seringkali terdapat data yang sama. Dengan menggunakan fungsi distinct, data yang sama hanya ditampilkan sekali saja sehingga tidak terjadi perulangan pada saat menampilkan data.

```
SELECT DISTINCT nama_kolom
FROM nama_tabel
```



String Function

- Fungsi String ialah fungsi yang digunakan untuk melakukan manipulasi data teks (string).
- Tujuan menggunakan fungsi string adalah untuk menghitung, menggabungkan, menguji, atau mengambil karakter yang diperlukan (bisa berposisikan di kiri, tengah, atau kanan dari teks) untuk kemudian dikombinasikan lagi dengan fungsi-fungsi yang lain sehingga memperoleh hasil yang diharapkan.
- Ada berbagai jenis fungsi string yang bisa digunakan dalam hal ini, seperti berikut:

Function	Return Type	Description	Example	Result
string || string	text	String concatenation	'Post' || 'greSQL'	PostgreSQL
string || non-string or non-string || string	text	String concatenation with one non-string input	'Value: ' || 42	Value: 42
char_length(string) or character_length(string)	int	Number of characters in string	char_length('jose')	4
lower(string)	text	Convert string to lowercase	lower('TOM')	tom
position(substring in string)	int	Location of specified substring	position('om' in 'Thomas')	3
substring(string [from int] [for int])	text	Extract substring	substring('Thomas' from 2 for 3)	hom
upper(string)	text	Convert string to uppercase	upper('tom')	TOM

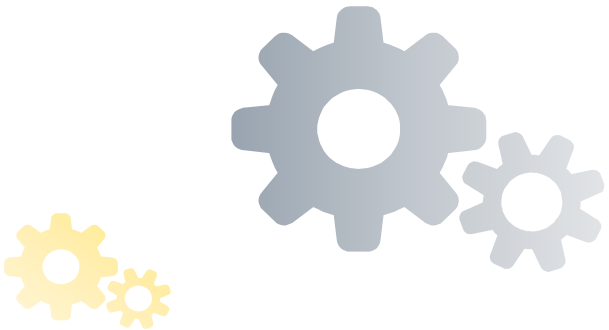
Aggregate Function

- Fungsi Agregat adalah sebuah kelompok perintah SQL yang bertujuan untuk menghasilkan sebuah data (*summarize*) pada sebuah tabel berdasarkan jumlah data atau record dalam sebuah kolom tabel. Fungsi agregat yang ada dalam SQL adalah SUM, MAX, MIN, AVG dan COUNT. Fungsi agregat ini bisa digunakan pada perintah SELECT untuk menampilkan data dengan memunculkan nilai tunggal.

Function	Argument Type(s)	Return Type	Description
`avg(`***expression***`)`	smallint, int, bigint, real, double precision, numeric, or interval	numeric for any integer-type argument, double precision for a floating-point argument, otherwise the same as the argument data type	the average (arithmetic mean) of all non-null input values
`count(*)`	any	bigint	number of input rows
`count(`***expression***`)`	any	bigint	number of input rows for which the value of *expression* is not null
`max(`***expression***`)`	any numeric, string, date/time, network, or enum type, or arrays of these types	same as argument type	maximum value of *expression* across all non-null input values
`min(`***expression***`)`	any numeric, string, date/time, network, or enum type, or arrays of these types	same as argument type	minimum value of *expression* across all non-null input values
`sum(`***expression***`)`	smallint, int, bigint, real, double precision, numeric, interval, or money	bigint for smallint or int arguments, numeric for bigint arguments, otherwise the same as the argument data type	sum of *expression* across all non-null input values

Case When Function

Fungsi Case When digunakan untuk menampilkan nilai tertentu dari beberapa barisan data dengan syarat-syarat atau kondisi yang kita berikan. Perintah Case When dapat digunakan untuk beberapa kondisi sekaligus dan tidak bisa berdiri sendiri melainkan harus disisipkan dalam perintah SELECT. Apabila kondisi urutan teratas telah ditemukan, maka fungsi ini akan berhenti membaca dan langsung mengeluarkan hasilnya. Jika tidak ada kondisi yang sesuai, fungsi ini menghasilkan nilai yang dimuat dalam klausa ELSE.



```
SELECT field1,
CASE
WHEN [field1] = 1 THEN 'Hasil‘
WHEN [field1] = 2 THEN 'Hasil'
WHEN ...
ELSE 'Hasil' END AS [alias]
FROM [table] ORDER BY [field]
```

Where

Perintah WHERE merupakan perintah dasar SQL yang di gunakan untuk mem-filter hasil Select dengan mengekstrak record yang memenuhi persyaratan tertentu. Singkatnya, query akan dijalankan jika memenuhi kondisi tertentu yang telah ditetapkan. Pada umumnya, fungsi ini juga ditambahkan And atau Or untuk membuat batasan-batasan kondisi yang lebih spesifik. And dan Or akan diproses seperti halnya pada logika matematika. Pada jenis field strings, perintah Where juga biasa dipadukan dengan equal (=) untuk memfilter nilai yang persis sama dengan yang diinginkan, atau menggunakan LIKE untuk mencari nilai yang mendekati.

```
SELECT kolom1, kolom2, …
FROM nama_tabel
WHERE kondisi;
```

Group By

Perintah Group by digunakan untuk menampilkan atau memilih sekumpulan data berdasarkan kelompok data tertentu. Pengelompokan dalam perintah Group By biasanya disertai oleh Aggregat Function. Hal penting yang perlu diperhatikan terkait dengan penggunaan Group By:

- Klausa SQL Group By digunakan dengan Select statement.
- Di dalam klausa SQL query Group By ditempatkan setelah klausa Where.
- Di dalam klausa SQL query Group By ditempatkan sebelum klausa Order by jika developer menggunakannya.

```
SELECT column1, function_name(column2)
FROM table_name
WHERE condition
GROUP BY column1, column2
ORDER BY column1, column2;
```

*)function_name adalah nama aggregat function yang digunakan seperti, SUM() , AVG().

Having

- Perintah Having digunakan untuk menggantikan Where ketika menggunakan Group By yang datanya di aggregasi.
- Secara umum Having digunakan setelah melakukan Group By. Seperti halnya pada Where, Having juga seringkali dipadukan dengan And dan Or.

```
SELECT nama_kolom
FROM nama_table
GROUP BY nama_kolom
HAVING kondisi
```

Order By

Perintah Order By digunakan untuk mengurutkan data baik dari satu kolom maupun lebih. Pengurutannya pun dapat dikombinasikan misalnya kolom pertama di urutkan dari kecil ke besar (ascending) dan kolom kedua dari besar ke kecil (descending). Secara default, pengurutannya akan bersifat ascending. Manfaat paling mendasar dari perintah Order by adalah data yang ditampilkan dapat disesuaikan urutannya berdasarkan keinginan.

```
SELECT *
FROM nama_tabel
ORDER BY kolom ASC
```

Fungsi Order By sebaiknya hanya digunakan ketika ada keperluan untuk mengurutkan data, sebab perintah ini cukup mengonsumsi memori komputer, apalagi untuk data dengan jumlah sangat besar. Fungsi ini juga dapat dipadukan dengan fungsi Limit untuk membuat query lebih cepat.

Limit

Fungsi limit digunakan untuk membatasi record yang ditampilkan pada saat membuat query di database SQL.

```
SELECT kolom_1, kolom_2
FROM nama_table
WHERE kondisi
ORDER BY nama_kolom [ASC|DESC]
LIMIT jumlah_record
```

Query Processing Order

Urutan proses query SQL bukan berdasarkan pernyataan yang paling atas (lebih dulu di tulis), akan tetapi dieksekusi dengan urutan sebagai berikut:

1. Getting Data (From, Join)
2. Row Filter (Where)
3. Grouping (Group by)
4. Group Filter (Having)
5. Return Expressions (Select)
6. Order & Paging (Order by & Limit / Offset)

DigitalSkola

THANKYOU