MySQL

1. DBMS adalah aplikasi untuk memanage data, contoh MySQL, Oracle, PostgreSQL, MonoDB, dll.
2. Yang biasanya kita gunakan adalah Rational database, ini printahnya di hampir sma d smua DBMS
3. Cara kerja DBMS: Database Client ->(dilakukan perintah SQL) -> DBMS -> Database file
4. Untuk membuat database = create database\_name;
5. Untuk menghapus database = drop database database\_name; (semua data bakal ilang) careful dan backup dulu
6. Untuk masuk/memilih database = use database\_name;
7. Tipe data itu per kolom contoh id (number) , nama (text): text itu bebas bisa angka atau huruf.
   1. Tipe data number:
      1. Integer (bilangan bulat)

| Type | Length in Bytes | Minimum Value (Signed) | Maximum Value (Signed) | Minimum Value (Unsigned) | Maximum Value (Unsigned) |
| --- | --- | --- | --- | --- | --- |
| TINYINT | 1 | -128 | 127 | 0 | 255 |
| SMALLINT | 2 | -32768 | 32767 | 0 | 65535 |
| MEDIUMINT | 3 | -8388608 | 8388607 to | 0 | 16777215 |
| INT | 4 | -2147483648 | 2147483647 | 0 | 4294967295 |
| BIGINT | 8 | -9223372036854775808 | 9223372036854775807 | 0 | 18446744073709551615 |

* + 1. Floating point (bilangan pecatan)
    2. Decimal

| DECIMAL | MIN | MAX |
| --- | --- | --- |
| DECIMAL (5,2) panjang 5 n koma 2 | -999.99 | 999.99 |
| DECIMAL (5,0) panjang 5 n koma 0 | -99999 | 99999 |
| DECIMAL (3,1) panjang 3 n koma 2 | -99.9 | 999 |
| DECIMAL (3) panjang 3 n koma 0 | -999 | 999 |

* + 1. Type Attribute

| TYPE(N} | Where N is an integer and display width of the type is up to N digits |
| --- | --- |
| ZEROFILL | The default padding of spaces is replaced with zeros. So, for a column INT(3) ZEROFILL, 7 is displayed as 007 |

1. Type data STRING
   1. CHAR ex: (CHAR)10. Memiliki max character 65535
   2. VARCHAR ex: (VARCHAR)10. Memiliki max character 65535
   3. Bisa di tentukan panjang maksimal nya dengan (value)

| Value | CHAR(4) | Storage Required | VARCHAR(4) | Storage Required |
| --- | --- | --- | --- | --- |
| ‘ ‘ | ‘ ' | 4 bytes | ‘ ’ | 1 bytes |
| ‘ab’ | ‘ab ‘ | 4 bytes | ‘ab’ | 3 bytes |
| ‘abcd’ | ‘abcd’ | 4 bytes | ‘abcd’ | 5 bytes |
| ‘abcdefgh’ | ‘abcd’ | 4 bytes | ‘abcd’ | 5 bytes |

1. Type string lainnya TEXT
   1. Tidak bisa ditentukan panjang maximum nya.
   2. Ada 4 jenis:
      1. TINYTEXT, maksimum 255 char (~256 bytes)
      2. TEXT, maks 65535 (~64 kb)
      3. MEDIUMTEXT, maks 16777215 (~16 MB)
      4. LONGTEXT, maks 4294967295 (~4GB)
2. Type data string ENUM
   1. bisa kita tentukan pilihannya:
      1. ENUM(Pria,Wanita) berarti hanya bisa menerima data Pria dan Wanita
3. Primary Key
   1. Sebuah table ideal nya memiliki Primary key yang kita tunjuk sebagai id table tersebut
   2. Ini adalah identitas untuk setiap baris data pada table tersebut
   3. Primary key harus unik, tidak boleh ada data dengan primary key yang sama
   4. Kita bisa menunjuk kolom yang akan kita jadikan Primary key
   5. Bisa di buat di multiple kolom
      1. Tetap disarankan hanya ada 1 kolom yang jadi primary key saja
      2. Kecuali ada kasus khusus, tabel yang berelasi MANY TO MANY
4. Fitur Alias berguna jika ingin mengubah nama kolom atau nama tabel ketika melakukan SELECT data:
   1. SELECT id as ‘Kode’ , name as ‘Nama’ , category as ‘Kategori’ , price as ‘Harga’ , quantity as ‘Stok’ FROM products; -> ini untuk kolom
   2. SELECT p.id as ‘Kode’ , p.name as ‘Nama’ , p.category as ‘Kategori’ , p.price as ‘Harga’ , p.quantity as ‘Stok’ FROM products as p; -> ini untuk alias di tabel
5. Operator yang bisa digunakan oleh WHERE clause
   1. Operator perbandingan

| Operator | Keterangan |
| --- | --- |
| = | Sama dengan |
| <> atau != | Tidak sama dengan |
| < | Kurang dari |
| <= | Kurang dari atau sama dengan |
| > | Lebih dari |
| >= | Lebih dari atau sama dengan |

1. LIKE Operator
   1. Bisa digunakan untuk mencari sebagian data dalam STRING
   2. Cocok untuk mencari sebuah kata dalam STRING
   3. Namun karena operasi LIKE sangat lambat, tidak disarankan jika data nya sudah besar
   4. Tidak case sensitive, jadi huruf besar atau kecil tidak masalah

| LIKE Operator | Hasil |
| --- | --- |
| LIKE ‘b%’ | String dengan awalan b |
| LIKE ‘%b’ | String dengan akhiran b |
| LIKE ‘%eko%’ | String berisi eko |
| NOT LIKE | Tidak LIKE |

1. Mencari data NULL operator
   1. IS NULL
   2. NOT NULL
2. BETWEEN Operator
   1. Untuk mencari nilai di antara 2 nilai
      1. WHERE price BETWEEN 9999 AND 20001
      2. WHERE price NOT BETWEEN 9999 AND 20001
3. IN Operator
   1. Untuk melakukan pencarian sebuah kolom dengan beberapa nilai
      1. WHERE category IN (‘Makanan’, ‘Minuman’);
      2. WHERE category NOT IN (‘Makanan’, ‘Minuman’);
4. ORDER BY Operator
   1. Digunakan untuk mengurutkan kolom yang dipilih jika kita menggunakan SELECT
   2. Jenis urutannya adalah ASC dan DESC
   3. Bisa digunakan untuk mengurutkan beberapa kolom
      1. ORDER BY price ASC/DESC;
5. LIMIT Clause
   1. Untuk membatasi jumlah data yang diambil dalam SQL SELECT
   2. Bisa untuk skip data yang tidak ingin dilihat
   3. Biasanya digunakan untuk paging di aplikasi
      1. SELECT \* FROM products WHERE price > 0 ORDER BY price LIMIT 2 (skip 2 data pertama), 2 (limit 2 data berikutnya);
6. Select DISTINCT data
   1. Digunakan untuk menghilangkan (tidak menunjukan) data2 yang duplikat saat menggunakan query SELECT. DISTINCT digunakan sebelum nama kolomnya.
      1. SELECT DISTINCT category FROM products.
7. Numeric Function
   1. Arithmetic Operator

| Name | Description |
| --- | --- |
| %, MOD | Modular Operator |
| \* | Multiplication Operator |
| + | Addition Operator |
| - | Minus Operator |
| - | Change the sign of the argument |
| / | Division Operator |
| DIV | Integer Division |

* + 1. SELECT 10 + 10 as Hasil;
    2. SELECT id, name, price DIV 1000 as ‘price in K’ FROM products;
  1. Mathematical Function
     1. Bisa dicari sendiri function nya di google karena banyak
     2. SELECT id, COS(price), SIN(price), TAN(price) FROM products;
     3. SELECT id, name, price FROM products WHERE price DIV 1000 > 15;

1. Auto increment in table making
   1. Digunakan untuk menandai bahwa suatu primary key datanya diisi secara otomatis dari angka terakhir +1
   2. Jadi tidak perlu selalu memasukan data primary key karena sudah otomatis terbuat
      1. CREATE TABLE admin (

id INT NOT NULL AUTO\_INCREMENT ,

first\_name VARCHAR(100),

PRIMARY KEY (id);

* 1. Bisa menggunakan SELECT LAST\_INSERT\_ID(); = ini digunakan untuk melihat ID terakhir yg di generate oleh auto\_increment

1. STRING function
   1. Bisa dicari sendiri function nya di google karena banyak
      1. SELECT id, LOWER(name) as ‘Name Lower’ FROM products;
      2. SELECT id, LENGTH(name) as ‘Name Length’ FROM products;
      3. SELECT id, UPPER(name) as ‘Name Upper’ FROM products;
2. Date and Time function
   1. Menambah kolom TIMESTAMP
      1. SELECT id,

Extract (YEAR FROM created\_at) AS Year,

Extract (MONTH FROM created\_at) AS Month

FROM products;

* + 1. SELECT id, YEAR(created\_at), MONTH(created\_at) FROM products;

1. Control Flow function
   1. Mirip seperti IF ELSE tapi tidak serumit di bahasa pemrograman
      1. SELECT id,

category

CASE category

WHEN ‘Makanan’ THEN ‘Enak’

WHEN ‘Minuman’ THEN ;Segar’

ELSE ‘Unidentified’

END AS ‘Category’

FROM products;

* + 1. SELECT id,

price,

IF(price <= 15000, ‘Murah’,IF (price <= 20000, ‘Mahal’, *selain itu* ‘Mahal Banget’)’) as ‘Mahal?’

FROM products;

* + 1. SELECT id, name, IFNULL(description, *ganti menjadi* ‘Kosong’) as ‘Kosong’

FROM products

1. Aggregate function
   1. Untuk melihat item paling mahal, murah, rata-rata, jumlah item dari suatu tabel.. dll smacam AVG, MIN, MAX, COUNT
      1. SELECT COUNT(id) as ‘Total Item’ FROM products;
      2. SELECT MAX(price) as ‘Item termahal’ FROM products;
      3. SELECT MIN(price) as ‘Item termurah’ FROM products;
      4. SELECT AVG(price) as ‘Rata-rata harga item’ FROM products;
      5. SELECT SUM(quantity) as ‘Total stock’ FROM products;
2. GROUP BY clause untuk melakukan grouping
   1. SELECT category,

COUNT(id) as ‘Total Product’, category *(untuk showing kolom kategori)*

FROM products GROUP BY category;

1. HAVING clause
   1. Untuk filtering data yang sudah d grouping
   2. Misal ingin menampilkan rata-rata harga per kategori, yang harganya diatas 10000
      1. SELECT category,

COUNT(id) as total

FROM products

GROUP BY category

HAVING total >1;

1. Unique constraint
   1. Constraint yg memastikan data kita tetap unique
   2. Kita jadi tidak bisa memasukan data yang sama karena akan d tolak MySQL
   3. Contoh saat membuat table
      1. CREATE TABLE customers (

email VARCHAR(100) NOT NULL,

UNIQUE KEY email\_unique (email)

)

* 1. Untuk menambah/menghapus bisa menggunakan
     1. ALTER TABLE customers

ADD CONSTRAINT email\_unique UNIQUE (email);

* + 1. ALTER TABLE customers

DROP CONSTRAINT email\_unique *(ini nama constraint nya)*;

1. Join , query ini untuk menggabungkan 2 buah table atau lebih. Ideal nya jangan lebih dari 5 karena berat
   1. SELECT \* FROM kolom JOIN products ON (kolom.id\_product = products.id);
   2. SELECT product.id, product.name, wishlist.description FROM wishlist JOIN product ON product.id = wishlist.id\_product;
   3. Melakukan multiple join
      1. SELECT \* FROM wishlist JOIN products ON products.id = wishlist.id\_product JOIN customers ON customers.id = wishlist.id\_customer;