B. Perintah SQL Dasar

SQL (*structure query language*) adalah bahasa khusus yang digunakan untuk mengelola basis data relasional. SQL sangat penting dalam pengelolaan data karena memungkinkan pengguna untuk membuat, memanipulasi, dan mengambil data dari basis data. Oleh karena itu, sangat penting untuk memahami SQL ketika ingin melakukan proses analisis data. Perhatikan penjelasan lebih lanjut pada materi berikut.

Kalimat Pemantik

Apakah Anda pernah belajar perintah SQL saat di SMP/MTS? Menurut Anda, bagaimana penerapan SQL dapat berguna dalam kehidupan sehari-hari?

1. Memahami Perintah SQL Dasar

SQL adalah bahasa yang digunakan untuk mengelola dan menjalankan perintah pada sistem manajemen basis data relasional atau *relational database management system* (RDBMS). RDBMS berfungsi mengedit, mengubah, menambah, memperbarui data, serta memainkan peran penting dalam analisis data besar dan visualisasi data. SQL berfungsi sebagai bahasa standar yang memungkinkan pengguna untuk berinteraksi dengan basis data.

Beberapa sistem RDBMS yang populer saat ini, seperti Oracle, Informix, MySQL, dan SQL Server mengadopsi SQL sebagai bahasa utama untuk pengelolaan data. Penggunaan SQL dalam RDBMS tidak hanya terbatas pada pengelolaan data, tetapi juga mencakup analisis dan visualisasi data. Dengan kemampuan ini, SQL menjadi alat yang esensial bagi para profesional di bidang teknologi informasi dan analisis data.

SQL memiliki beberapa sifat unik, salah satunya bersifat *noncase sensitive* yang artinya tidak membedakan huruf besar dan kecil pada sintaksnya, meskipun konvensi penulisan umumnya menggunakan huruf kapital untuk perintah agar lebih konsisten dan mudah dibaca. Struktur pernyataan dalam SQL bergantung pada penulisan baris teks yang tepat karena format dan susunan perintah dapat memengaruhi hasil eksekusi. Selain itu, SQL juga berhubungan erat dengan konsep kalkulus dan aljabar yang menjadi dasar dalam pengolahan dan pengambilan data.

Ketika pengguna meng-*input* perintah SQL, sistem RDBMS akan melakukan serangkaian proses untuk menentukan cara terbaik dalam mengeksekusi

permintaan tersebut. Proses ini melibatkan beberapa komponen, seperti *optimization engine*, *query engine*, dan *query*.

dispatcher. Komponen tersebut bekerja sama untuk memastikan efisiensi dan keakuratan dalam pengolahan data.

2. Jenis Perintah dalam SQL

SQL menyediakan berbagai perintah yang memungkinkan pengguna untuk melakukan operasi pada data, termasuk membuat, membaca, memperbarui, dan menghapus data (operasi yang dikenal sebagai CRUD). Berikut adalah tiga jenis perintah SQL yang umum digunakan.

a. Data definition language (DDL)

Perintah DDL berguna untuk mendefinisikan struktur basis data yang akan dibuat. Perintah DDL terdiri atas beberapa perintah utama berikut.

1) CREATE TABLE

Perintah ini berfungsi membuat tabel baru dalam basis data. Sebagai contoh, untuk membuat tabel bernama *PesertaDidik* dengan empat kolom, yaitu ID, Nama, Umur, dan Kota perintah yang digunakan adalah sebagai berikut.

```
CREATE TABLE PesertaDidik (
    ID INT PRIMARY KEY,
    Nama VARCHAR(50),
    Umur INT,
    Kota VARCHAR(50)
);
```

2) ALTER TABLE

Perintah ini berfungsi mengubah struktur tabel yang sudah ada, seperti menambahkan kolom dalam tabel. Sebagai contoh, untuk menambahkan kolom *Email* pada tabel *PesertaDidik*, perintah yang digunakan adalah sebagai berikut.

```
ALTER TABLE PesertaDidik ADD Email VARCHAR(100);
```

3) DROP TABLE

Perintah ini berfungsi menghapus tabel dari basis data. Sebagai contoh, untuk menghapus tabel *PesertaDidik*, perintah yang digunakan adalah sebagai berikut.

```
DROP TABLE PesertaDidik;
```

b. Data manipulation language (DML)

Perintah DML berfungsi memanipulasi data pada tabel yang ada di *database*. Proses manipulasi yang dapat dilakukan, meliputi penambahan, penghapusan, pengubahan, dan pengambilan data. Berikut adalah beberapa perintah yang termasuk dalam DML.

1) INSERT INTO

Perintah ini berfungsi menambahkan data baru ke dalam tabel. Sebagai contoh, jika ingin menambahkan data baru ke dalam tabel *PesertaDidik*, perintah yang digunakan adalah sebagai berikut.

```
INSERT INTO PesertaDidik (ID, Nama, Umur, Kota)
VALUES (1, 'Andi', 17, 'Jakarta');
```

2) SELECT

Perintah ini berfungsi mengambil dan menampilkan data dari tabel. Sebagai contoh, jika ingin menampilkan semua kolom dari tabel *PesertaDidik*, kode yang digunakan adalah sebagai berikut.

```
SELECT * FROM PesertaDidik;
```

3) UPDATE

Perintah ini berfungsi memperbarui data yang sudah ada di dalam tabel. Sebagai contoh, jika ingin memperbarui data umur peserta didik dengan ID 1 pada tabel *PesertaDidik*, perintah yang digunakan adalah sebagai berikut.

```
UPDATE PesertaDidik SET Umur = 18 WHERE ID = 1;
```

4) DELETE

Perintah ini berfungsi menghapus data dari tabel. Sebagai contoh, jika ingin menghapus data peserta didik dengan ID 1 dari tabel *PesertaDidik*, perintah yang digunakan adalah sebagai berikut.

```
DELETE FROM PesertaDidik WHERE ID = 1;
```

c. Data query language (DQL)

Perintah DQL berfokus pada pengambilan data dari *database* berdasarkan kriteria tertentu. Berikut adalah beberapa contoh perintah DQL.

1) WHERE

Perintah ini berfungsi mengambil data dengan kondisi tertentu. Sebagai contoh, untuk mengambil data semua peserta didik yang berasal dari Jakarta, perintah yang digunakan adalah sebagai berikut.

```
SELECT * FROM PesertaDidik WHERE Kota = 'Jakarta';
```

2) ORDER BY

Perintah ini berfungsi mengurutkan hasil *query* berdasarkan kolom tertentu. Sebagai contoh, untuk mengurutkan data berdasarkan kolom *Umur* dari yang terkecil hingga terbesar, perintah yang digunakan adalah sebagai berikut.

```
SELECT * FROM PesertaDidik ORDER BY Umur ASC;
```

3) GROUP BY

Perintah ini berfungsi mengelompokkan hasil berdasarkan satu atau lebih kolom dan menghitung agregat. Sebagai contoh, untuk mengambil data peserta didik dan mengelompokkannya berdasarkan kota serta menghitung jumlah peserta didik di tiap kota, perintah yang digunakan adalah sebagai berikut.

```
SELECT Kota, COUNT(*) AS Jumlah PesertaDidik
FROM PesertaDidik GROUP BY Kota;
```

4) HAVING

Perintah ini berfungsi mengevaluasi kondisi pada kelompok yang dihasilkan oleh perintah GROUP BY. Sebagai contoh, untuk mengambil rata-rata umur peserta didik per kota, tetapi hanya menampilkan kota dengan rata-rata umur lebih dari 17, perintah yang digunakan adalah sebagai berikut.

```
SELECT Kota, AVG(Umur) AS RataRataUmur FROM
PesertaDidik GROUP BY Kota HAVING AVG(Umur) > 17;
```