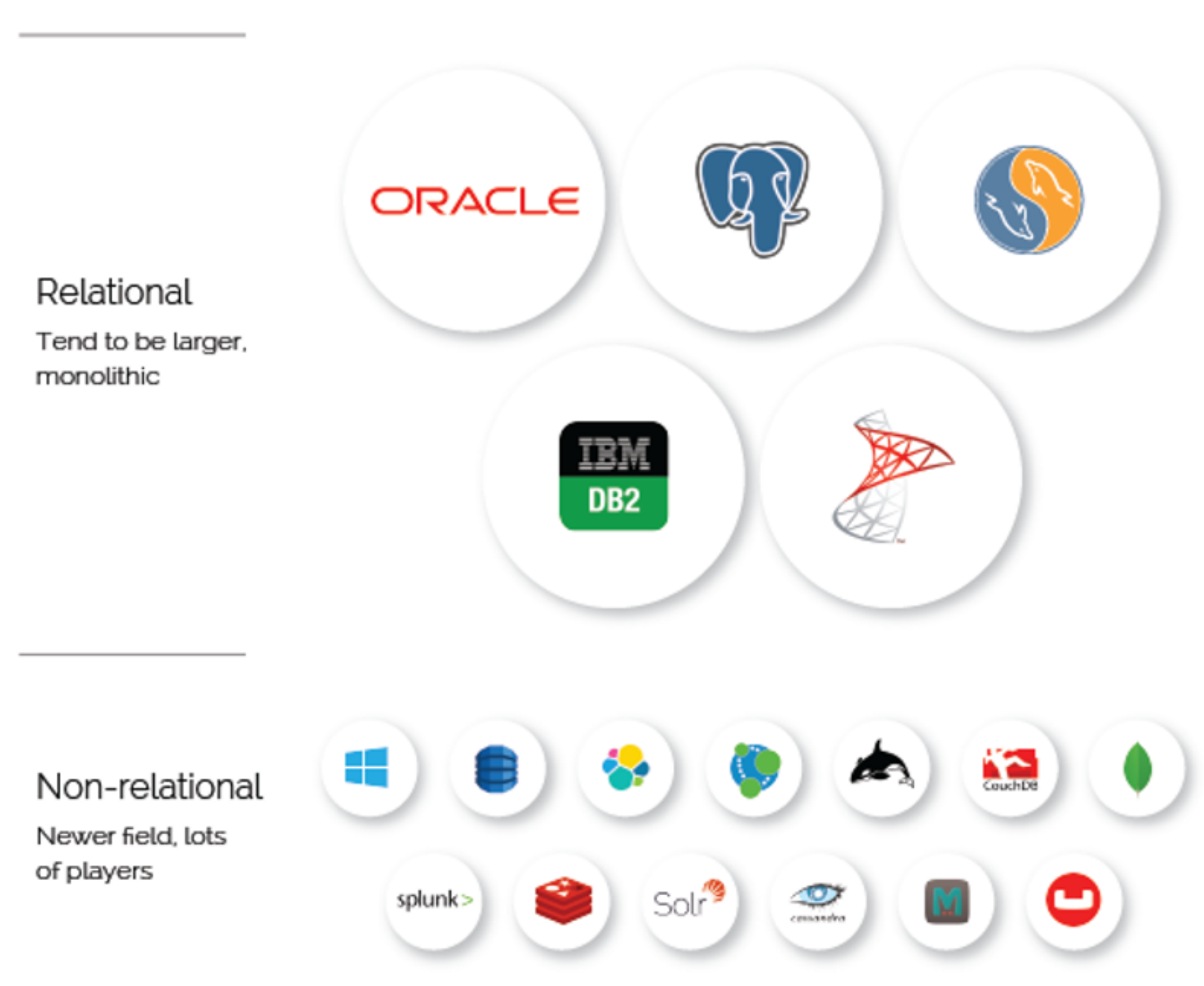
Introduction to SQL (Part 1)

Database Concept

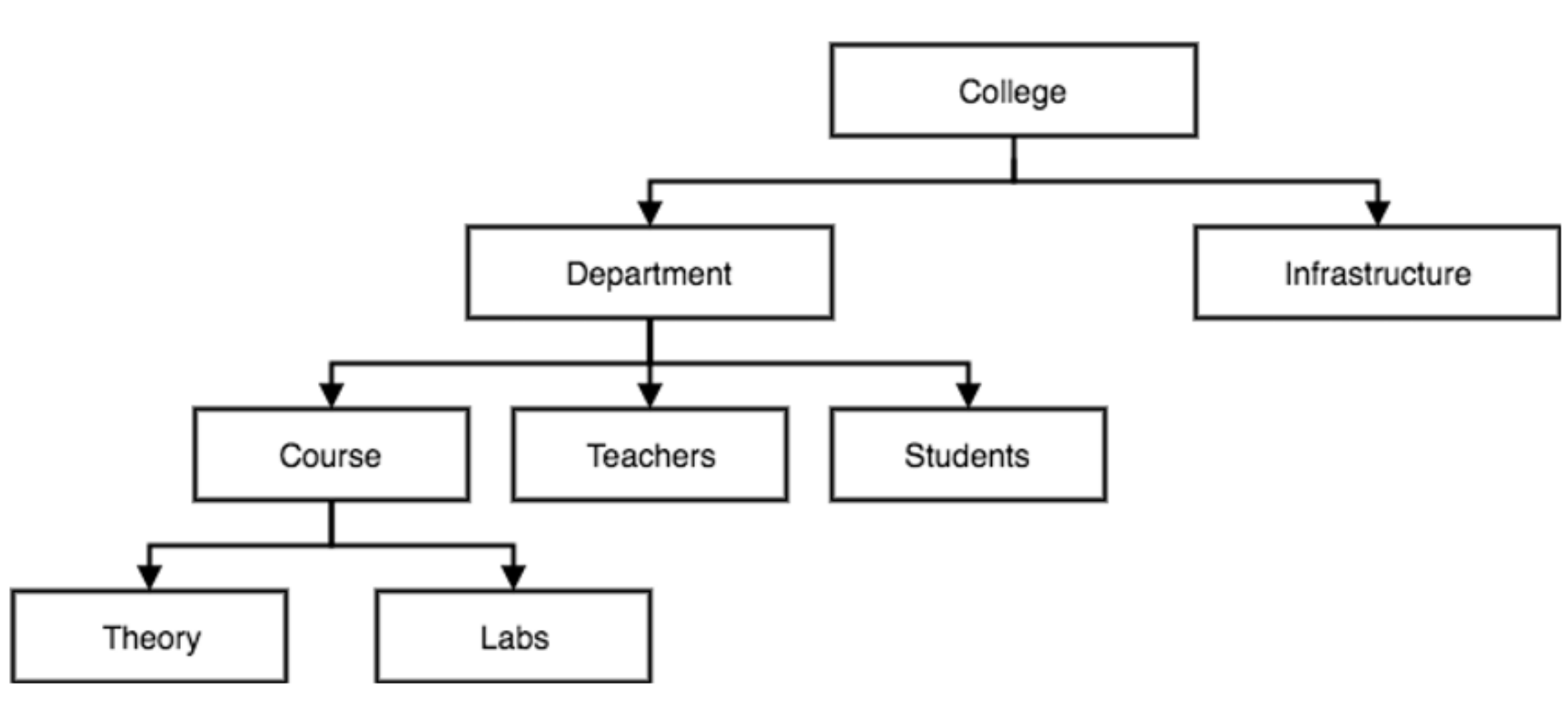
Database adalah susunan record data operasional lengkap dari suatu organisasi atau perusahaan, yang diorganisir dan disimpan secara terintegrasi dengan menggunakan metode tertentu dalam komputer sehingga mampu memenuhi informasi yang optimal yang dibutuhkan oleh para pengguna. Database sangat penting dalam kehidupan dan pada zaman modern yang sekarang ini karena database merupakan landasan bagi pembuatan dan pengembangan program aplikasi.

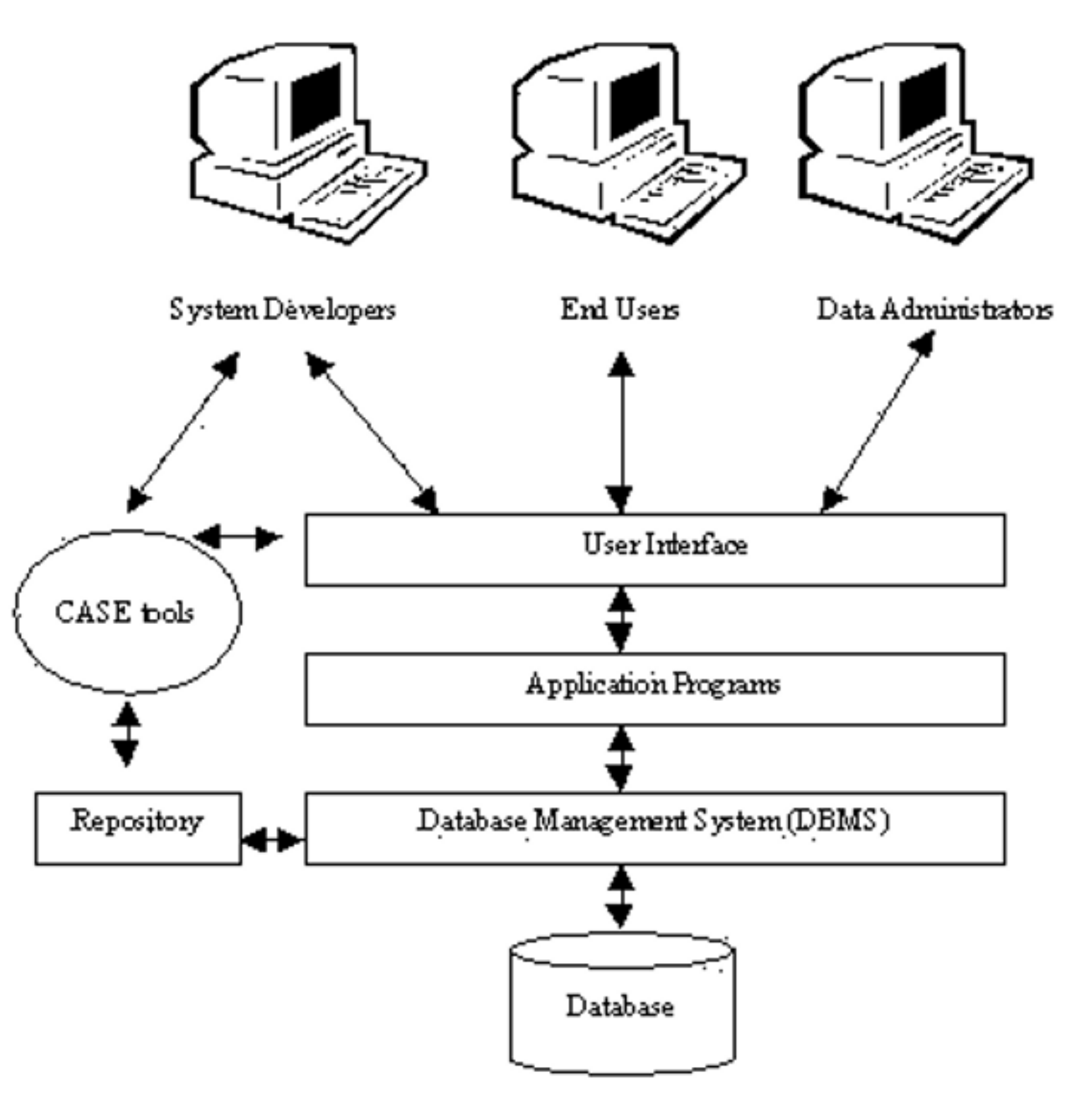
Produk Software



RDBMS (Relational Database Management System)

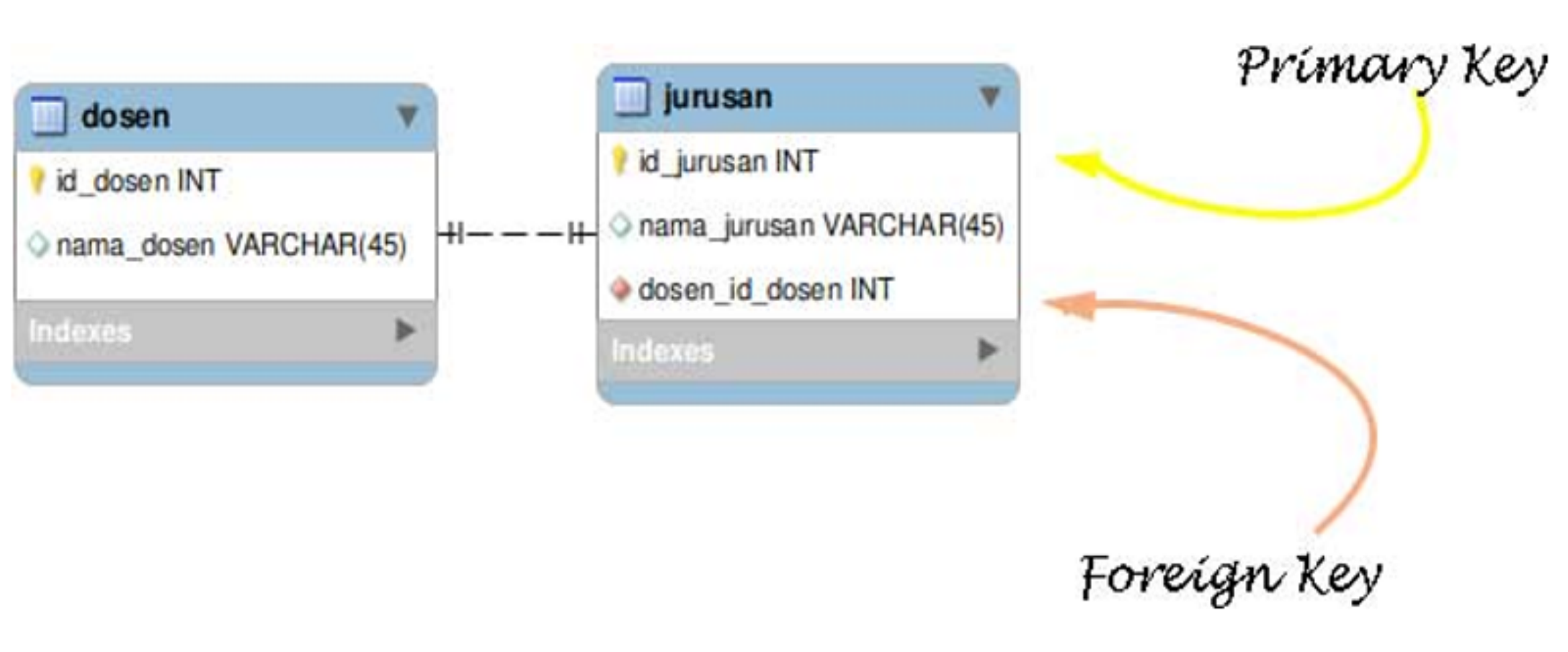
RDBMS adalah kependekan dari Relational Database Management System. RDBMS adalah program yang melayani sistem basis data yang entitas utamanya terdiri dari tabel-tabel yang mempunyai relasi dari satu tabel ke tabel yang lain. Dengan relational database ini maka data akan secara konsisten disimpan di suatu tabel, kemudian tabel lain yang membutuhkan data lainnya tinggal menghubungkan melalui foreign key.





Pada sebuah database, relasi dihubungkan dengan dua tabel yang dihubungkan melalui kolom foreign key pada tabel pertama dengan primary key tabel kedua.

Berdasarkan gambar berikut terdapat 2 tabel yaitu tabel dosen dan tabel jurusan yang memiliki primary key dan foreign key. Pada tabel jurusan terdapat primary key id_jurusan dan foreign key dosen_id_dosen yang berelasi ke tabel dosen. Sedangkan isi dari foreign key dosen_id_dosen adalah id_dosen pada tabel dosen. Foreign key yang ada pada tabel jurusan digunakan untuk menghubungkan tabel dosen dengan tabel jurusan.



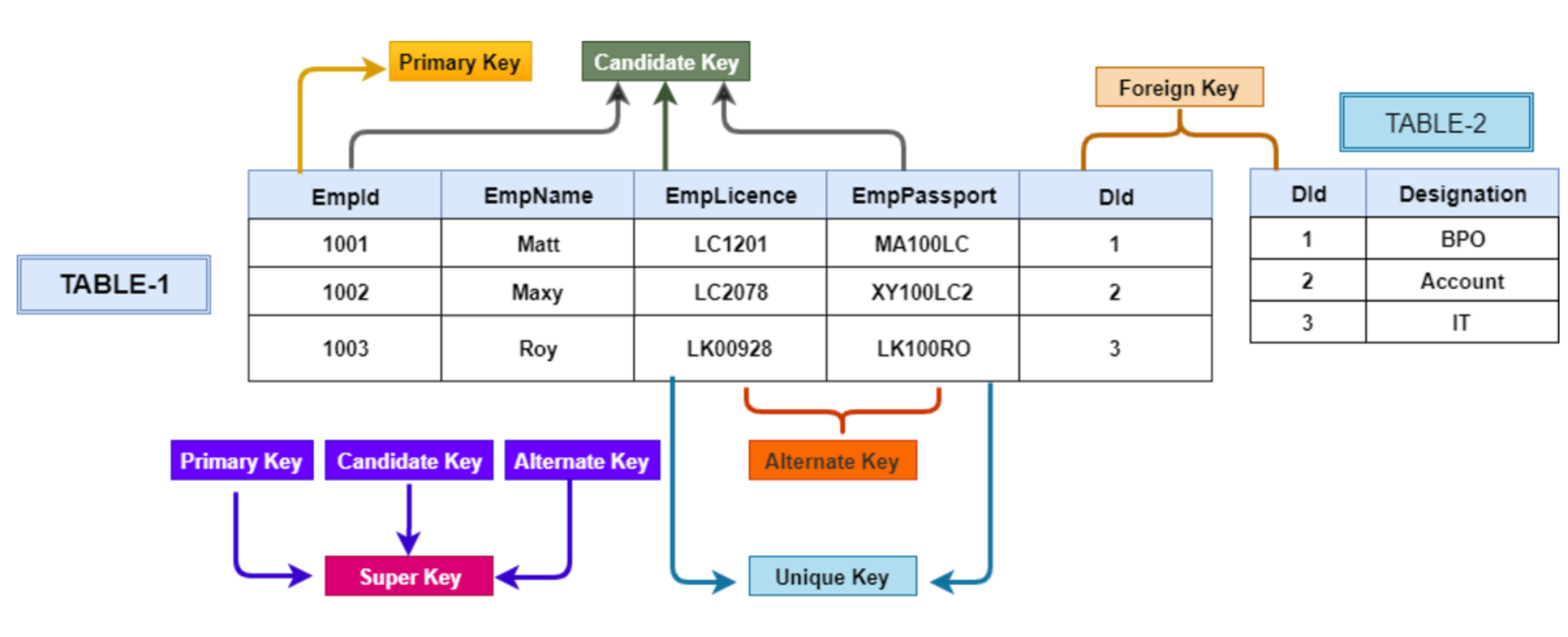


TABLE-1

EmpId	EmpName	EmpLicence	EmpPassport	DId
1001	Matt	LC1201	MA100LC	1
1002	Maxy	LC2078	XY100LC2	2
1003	Roy	LK00928	LK100RO	3

TABLE-2

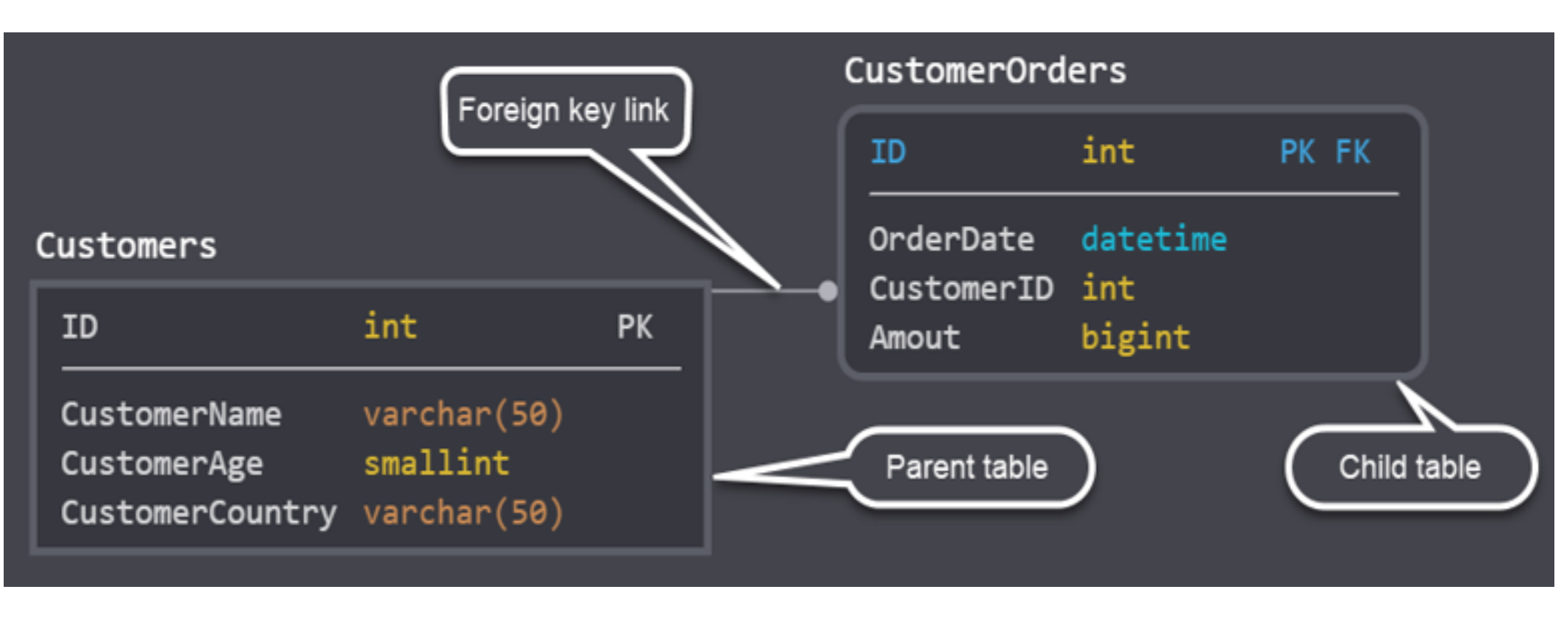
DId	Designation
1	BPO
2	Account
3	IT

Beberapa hal yang harus kamu ketahui tentang foreign key & primary key:

- Harus unik
- Tabel hanya boleh memiliki satu primary key, namun dalam beberapa kasus boleh lebih dari 1 primary key (composite key)
- Tabel boleh memiliki lebih dari satu foreign key
- Foreign key digunakan untuk membuat relasi antar tabel

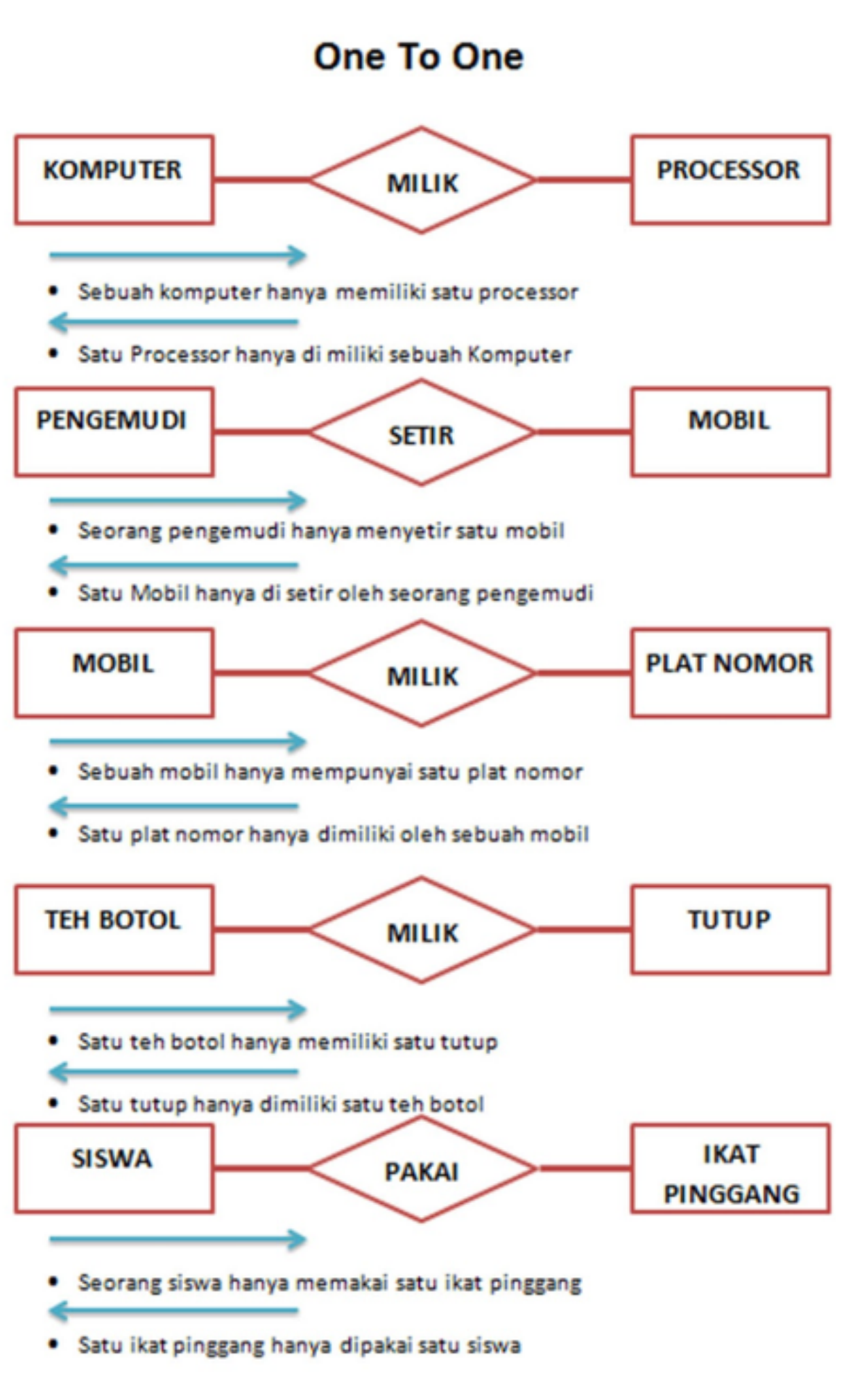
Ada beberapa jenis relasi database, yang akan dibahas adalah:

- One to One
- One to Many
- Many to Many



One to One

Setiap baris data pada tabel pertama dihubungkan hanya ke satu baris data pada tabel kedua. Hubungan antara file pertama dan file kedua adalah satu berbanding satu.

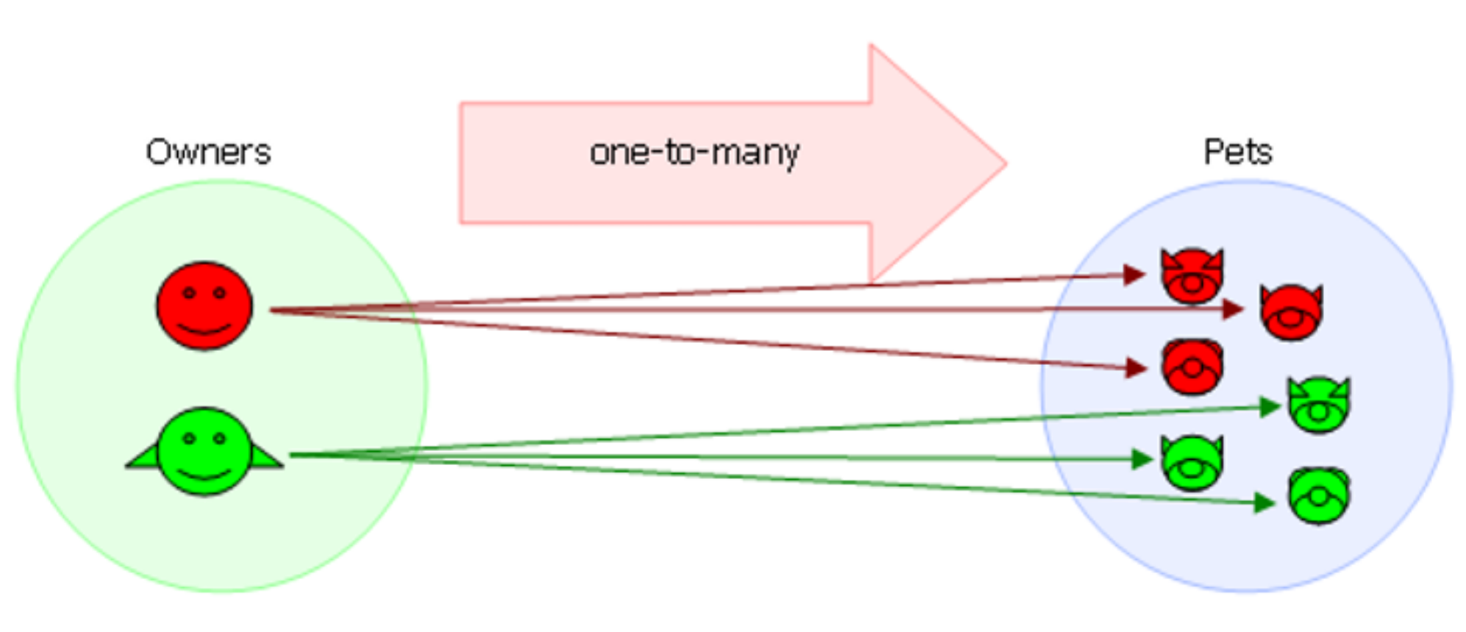


- Sebuah komputer hanya memiliki satu processor
- Satu Processor hanya di miliki sebuah Komputer
- Seorang pengemudi hanya menyetir satu mobil
- Satu Mobil hanya di setir oleh seorang pengemudi
- Sebuah mobil hanya mempunyai satu plat nomor
- Satu plat nomor hanya dimiliki oleh sebuah mobil
- Satu teh botol hanya memiliki satu tutup
- Satu tutup hanya dimiliki satu teh botol
- Seorang siswa hanya memakai satu ikat pinggang
- Satu ikat pinggang hanya dipakai satu siswa

One to Many

Setiap baris data dari tabel pertama dapat dihubungkan ke satu baris atau lebih data pada tabel kedua. Hubungan antara file pertama dan file kedua adalah satu berbanding banyak atau banyak berbanding satu. Contoh :

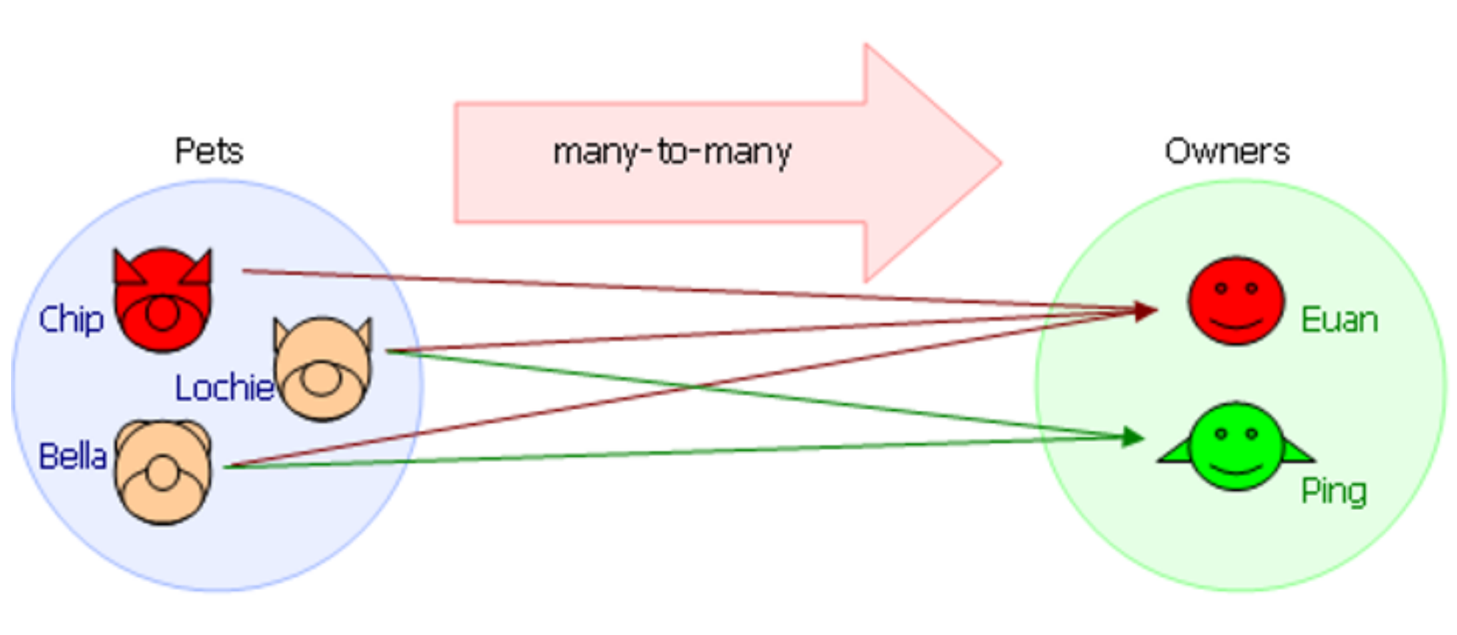
- Dalam satu perusahaan mempekerjakan banyak pegawai.
- Satu bagian memperkerjakan banyak pegawai, Satu pegawai kerja dalam satu bagian.



Many to Many

Satu baris atau lebih data pada tabel pertama bisa dihubungkan ke satu atau lebih baris data pada tabel kedua. Artinya ada banyak baris di tabel satu dan tabel dua yang saling berhubungan satu sama lain. Hubungan tabel pertama dan tabel kedua adalah banyak berbanding banyak. Contoh :

- Dalam universitas seorang mahasiswa dapat mengambil banyak mata kuliah
- Satu mahasiswa banyak mengambil banyak mata kuliah dan satu mata kuliah diambil banyak mahasiswa.



SQL Concepts

Structured Query Language (SQL) adalah sistem manajemen database relasional (RDBMS) yang dirancang untuk aplikasi dengan arsitektur client/server. Istilah client, server, dan client/server dapat digunakan untuk merujuk kepada konsep yang sangat umum atau hal yang spesifik dari perangkat keras atau perangkat lunak.

Berikut ini adalah beberapa kegunaan dari SQL diantaranya :

- SQL memungkinkan anda mengakses dan memanipulasi database.
- SQL dapat mengeksekusi query terhadap database
- SQL dapat mengambil data dari database
- SQL dapat menyisipkan catatan dalam database
- SQL dapat memperbarui catatan dalam database

Pada umumnya terdapat 3 (tiga) jenis perintah SQL yang bisa digunakan oleh SQL, yaitu:

1. DDL (Data Definition Language)
2. DML (Data Manipulation Language)
3. DCL (Data Control Language).

DDL atau Data Definition Language

DDL merupakan perintah SQL yang berhubungan dengan pendefinisian suatu struktur database, dalam hal ini database dan table. Beberapa perintah dasar yang termasuk DDL ini antara lain :

- CREATE : Seperti namanya, perintah create digunakan untuk membuat sesuatu, dalam hal ini adalah database dan table.
- ALTER : Perintah alter digunakan untuk merubah struktur atau mengubah informasi. Perintah alter bisa digunakan untuk database ataupun table.
- RENAME : Perintah rename biasanya digunakan untuk mengubah nama table, apabila sebuah tabel ingin diganti namanya.
- DROP : Perintah drop digunakan untuk menghapus, maka apabila menggunakan perintah ini harus berhati-hati karena drop dapat mengakses database, tabel, kolom, index, procedure dan yang lainnya.

DML atau Data Manipulation Language

DML merupakan perintah SQL yang berhubungan dengan manipulasi atau pengolahan data atau record dalam table. Perintah SQL yang termasuk dalam DML antara lain:

- SELECT : Perintah select digunakan untuk menampilkan data-data yang ada didalam tabel pada suatu database.
- INSERT : Perintah insert digunakan untuk menambahkan data pada tabel yang terdapat didalam database.
- UPDATE : Update digunakan untuk mengubah data, atau memodifikasi data yang terdapat didalam tabel.
- DELETE : Perintah delete digunakan untuk menghapus data atau record dalam table.

DCL atau Data Control Language

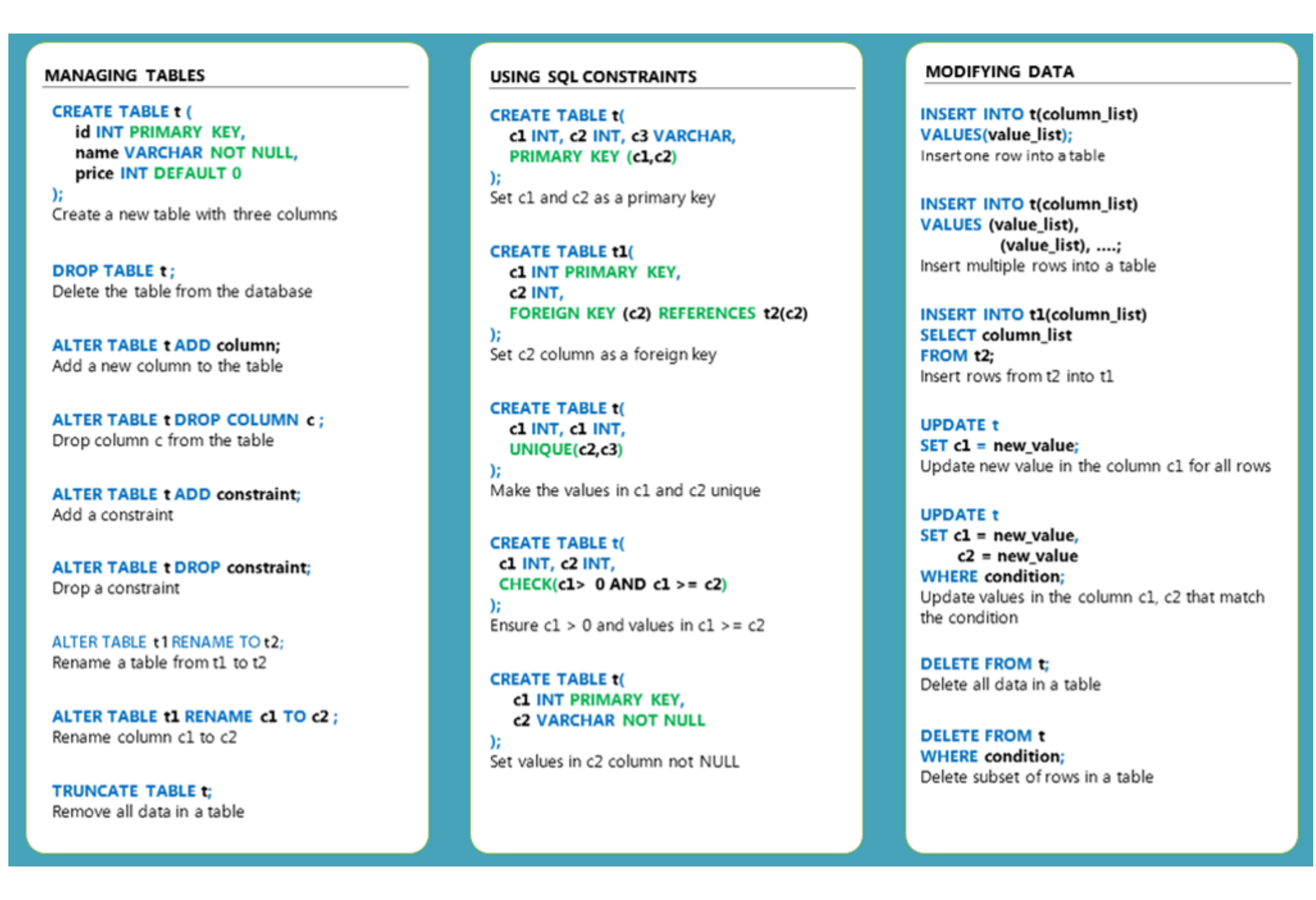
DCL merupakan perintah SQL yang berhubungan dengan manipulasi user dan hak akses (privileges). Perintah SQL yang termasuk dalam DCL antara lain :

- GRANT : Perintah grant digunakan untuk memberikan hak akses atau izin pada user di database untuk dapat mengakses database tersebut. Selain itu perintah grant juga dapat digunakan untuk menambahkan user atau pengguna baru di DBMS.
- REVOKE : Perintah revoke adalah kebalikan dari perintah grant, perintah revoke digunakan untuk menghapus atau atau mencabut izin hak akses.

Creating Database Engine

```
CREATE DATABASE databasename;
```

```
CREATE DATABASE testDB;
```



MANAGING TABLES

```
CREATE TABLE t (
    id INT PRIMARY KEY,
    name VARCHAR NOT NULL,
    price INT DEFAULT 0
);
```
Create a new table with three columns

```
DROP TABLE t;
```
Delete the table from the database

```
ALTER TABLE t ADD column;
```
Add a new column to the table

```
ALTER TABLE t DROP COLUMN c ;
```
Drop column c from the table

```
ALTER TABLE t ADD constraint;
```
Add a constraint

```
ALTER TABLE t DROP constraint;
```
Drop a constraint

```
ALTER TABLE t1 RENAME TO t2;
```
Rename a table from t1 to t2

```
ALTER TABLE t1 RENAME c1 TO c2 ;
```
Rename column c1 to c2

```
TRUNCATE TABLE t;
```
Remove all data in a table

USING SQL CONSTRAINTS

```
CREATE TABLE t(
   c1 INT, c2 INT, c3 VARCHAR,
   PRIMARY KEY (c1,c2)
);
```
Set c1 and c2 as a primary key

```
CREATE TABLE t1(
   c1 INT PRIMARY KEY,
   c2 INT,
   FOREIGN KEY (c2) REFERENCES t2(c2)
);
```
Set c2 column as a foreign key

```
CREATE TABLE t(
   c1 INT, c1 INT,
   UNIQUE(c2,c3)
);
```
Make the values in c1 and c2 unique

```
CREATE TABLE t(
  c1 INT, c2 INT,
  CHECK(c1> 0 AND c1 >= c2)
);
```
Ensure c1 > 0 and values in c1 >= c2

```
CREATE TABLE t(
   c1 INT PRIMARY KEY,
   c2 VARCHAR NOT NULL
);
```
Set values in c2 column not NULL

MODIFYING DATA

```
INSERT INTO t(column_list)
VALUES(value_list);
```
Insert one row into a table

```
INSERT INTO t(column_list)
VALUES (value_list),
         (value_list), ....;
```
Insert multiple rows into a table

```
INSERT INTO t1(column_list)
SELECT column_list
FROM t2;
```
Insert rows from t2 into t1

```
UPDATE t
SET c1 = new_value;
```
Update new value in the column c1 for all rows

```
UPDATE t
SET c1 = new_value,
     c2 = new_value
WHERE condition;
```
Update values in the column c1, c2 that match the condition

```
DELETE FROM t;
```
Delete all data in a table

```
DELETE FROM t
WHERE condition;
```
Delete subset of rows in a table

Querying Single Table

Fetch all columns from the `country` table:

```
SELECT *
FROM country;
```

Fetch id and name columns from the city table:

```
SELECT id, name
FROM city;
```

Fetch city names sorted by the `rating` column in the default ASCending order:

```
SELECT name
FROM city
ORDER BY rating [ASC];
```

Fetch city names sorted by the `rating` column in the DESCending order:

```
SELECT name
FROM city
ORDER BY rating DESC;
```

Aliases

COLUMNS

```
SELECT name AS city_name
FROM city;
```

TABLES

```
SELECT co.name, ci.name
FROM city AS ci
JOIN country AS co
  ON ci.country_id = co.id;
```

Filtering the output

COMPARISON OPERATORS

Fetch names of cities that have a rating above 3:

```
SELECT name
FROM city
WHERE rating > 3;
```

Fetch names of cities that are neither Berlin nor Madrid:

```
SELECT name
FROM city
WHERE name != 'Berlin'
  AND name != 'Madrid';
```

TEXT OPERATORS

Fetch names of cities that start with a 'P' or end with an 's':

```
SELECT name
FROM city
WHERE name LIKE 'P%'
  OR name LIKE '%s';
```

Fetch names of cities that start with any letter followed by 'ublin' (like Dublin in Ireland or Lublin in Poland):

```
SELECT name
FROM city
WHERE name LIKE '_ublin';
```

OTHER OPERATORS

Fetch names of cities that have a population between 500K and 5M:

```
SELECT name
FROM city
WHERE population BETWEEN 500000 AND 5000000;
```

Fetch names of cities that don't miss a rating value:

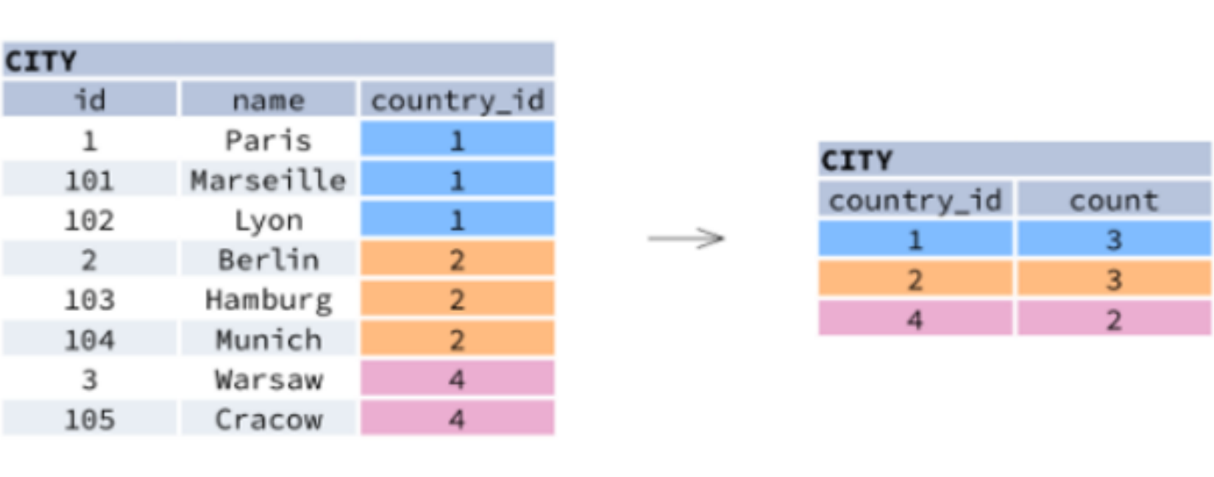
```
SELECT name
FROM city
WHERE rating IS NOT NULL;
```

Fetch names of cities that are in countries with IDs 1, 4, 7, or 8:

```
SELECT name
FROM city
WHERE country_id IN (1, 4, 7, 8);
```

Aggregation and Grouping

`GROUP BY` **groups** together rows that have the same values in specified columns. It computes summaries (aggregates) for each unique combination of values.



CITY

id	name	country_id
1	Paris	1
101	Marseille	1
102	Lyon	1
2	Berlin	2
103	Hamburg	2
104	Munich	2
3	Warsaw	4
105	Cracow	4

→

CITY

country_id	count
1	3
2	3
4	2

AGGREGATE FUNCTIONS

- `avg(expr)` – average value for rows within the group
- `count(expr)` – count of values for rows within the group
- `max(expr)` – maximum value within the group
- `min(expr)` – minimum value within the group
- `sum(expr)` – sum of values within the group

EXAMPLE QUERIES

Find out the number of cities:

```
SELECT COUNT(*)
FROM city;
```

Find out the number of cities with non-null ratings:

```
SELECT COUNT(rating)
FROM city;
```

Find out the number of distinctive country values:

```
SELECT COUNT(DISTINCT country_id)
FROM city;
```

Find out the smallest and the greatest country populations:

```
SELECT MIN(population), MAX(population)
FROM country;
```

Find out the total population of cities in respective countries:

```
SELECT country_id, SUM(population)
FROM city
GROUP BY country_id;
```

Find out the average rating for cities in respective countries if the average is above 3.0:

```
SELECT country_id, AVG(rating)
FROM city
GROUP BY country_id
HAVING AVG(rating) > 3.0;
```