Diberikan skema basis data berikut ini. Atribut dengan garis bawah adalah *primary key*, sedangkan atribut tercetak *miring* adalah *foreign key* (ke atribut *primary key* dengan nama sama):

SUPPLIER = (SID, SNAME, SADDRESS, SCITY, RATING, BALANCE)
PRODUCT = (PID, PNAME, *CID*, WEIGHT)
PRODUCT_CATEGORY = (CID, CNAME, DESCRIPTION)
SUPPLY = (ORDERDATE, *PID*, *SID*, QUANTITY, PRICE, RECEIVEDATE)

Berdasarkan hasil analisis terhadap *workload* sistem yang telah berjalan dan data yang ada di basis data, diperoleh fakta berikut ini.

- Terdapat 1000 *record* pada SUPPLIER, 12000 pada PRODUCT, 150000 pada SUPPLY, dan 100 pada PRODUCT_CATEGORY.
- 90% akses terhadap produk selalu membutuhkan nama kategori dari produk tersebut.
- 75% akses terhadap *supply* hanya mengakses data pesanan dari *supplier* yang belum diterima saja. Setiap saat, rata-rata hanya ada 2000 hingga 3000 pesanan dari *supplier* yang belum diterima.
- Atribut Alamat, Kota, dan Rating dari *supplier* hanya diakses setiap awal bulan, pada saat akan membuat laporan kegiatan pada bulan sebelumnya. Atribut Balance sering sekali berubah, yaitu setiap kali ada pesanan ke *supplier* tersebut.
- Pencarian data supplier dan product sangat sering dilakukan berdasarkan nama.
- Untuk memenuhi kebutuhan dari pihak eksekutif, query untuk mendapatkan total harga terhutang dari seluruh pesanan yang belum diterima, baik per *supplier* maupun per tanggal pesanan, sering dijalankan.

Tugas Anda:

- Berikan usulan schema tuning untuk meningkatkan kinerja sistem basis data di atas, termasuk kemungkinan untuk memanfaatkan materialized view dan kebutuhan akan indeks. Indeks yang tidak perlu dituliskan adalah indeks yang berkaitan dengan primary key. Untuk setiap usulan:
- Tuliskan jenis tuning-nya dan jelaskan.
- Tuliskan skema baru yang dihasilkan (jika ada).
- Tuliskan kebutuhan proses tambahan di basis data untuk tetap menjaga konsistensi data (jika ada).
- Jika perlu index, tuliskan dengan jelas relasi dan atribut indeks dan jenis indeks yang dibutuhkan.
- Untuk setiap materialized view yang Anda usulkan, tuliskan perintah create view untuk mendapatkannya.
- Di akhir jawaban, tuliskan kembali seluruh skema basis data final yang Anda usulkan, termasuk skema materialized view (jika ada).

Kasus 1

Untuk mengatasi permasalahan “90% akses terhadap produk selalu membutuhkan nama kategori dari produk tersebut”, jenis tuning yang digunakan adalah **denormalisasi (adding redundant columns)**. Diperlukan denormalisasi pada tabel PRODUCT dengan menambahkan CNAME dari tabel PRODUCT_CATEGORY.

Skema barunya adalah

PRODUCT = (PID, PNAME, CID, **CNAME**, WEIGHT)

Untuk PID yang sama, nilai atribut CNAME pada tabel PRODUCT yang baru harus dipastikan sama dengan nilai atribut CNAME pada tabel PRODUCT_CATEGORY.

Kasus 2

Untuk mengatasi permasalahan “75% akses terhadap supply hanya mengakses data pesanan dari supplier yang belum diterima saja. Setiap saat, rata-rata hanya ada 2000 hingga 3000 pesanan dari supplier yang belum diterima.”, jenis tuning yang digunakan adalah **horizontal splitting**. Diperlukan horizontal splitting pada tabel SUPPLY berdasarkan nilai atribut RECEIVEDATE. Tabel SUPPLY_RECEIVED berisi data SUPPLY dengan RECEIVEDATE not NULL. Tabel SUPPLY_NOTRECEIVED berisi data SUPPLY dengan RECEIVEDATE is NULL.

Skema barunya adalah

SUPPLY_RECEIVED = (ORDERDATE, ***PID, SID***, QUANTITY, PRICE, RECEIVEDATE)

SUPPLY_NOTRECEIVED = (ORDERDATE, ***PID, SID***, QUANTITY, PRICE, RECEIVEDATE)

Data pada SUPPLY_RECEIVED harus disjoin dengan data SUPPLY_NOTRECEIVE (berdasarkan primary key). Nilai RECEIVEDATE pada tabel SUPPLY_RECEIVED harus NOT NULL, sedangkan data RECEIVEDATE pada tabel SUPPLY_NOTRECEIVED harus NULL. Jika pesanan dari supplier yang belum diterima pada tabel SUPPLY_NOTRECEIVED berubah status menjadi diterima, namun baris data tersebut harus dipindahkan dari tabel SUPPLY_NOTRECEIVED ke SUPPLY_RECEIVED.

Kasus 3

Untuk mengatasi permasalahan “Atribut Alamat, Kota, dan Rating dari supplier hanya diakses setiap awal bulan, pada saat akan membuat laporan kegiatan pada bulan sebelumnya. Atribut Balance sering sekali berubah, yaitu setiap kali ada pesanan ke supplier tersebut.”, jenis tuning yang digunakan adalah **vertical splitting**. Atribut Alamat, Kota, dan Rating dari supplier hanya diakses setiap awal bulan. Pada saat akan membuat laporan kegiatan pada bulan sebelumnya, atribut Balance sering berubah saat setiap kali ada pesanan ke supplier tersebut sehingga perlu dilakukan vertical splitting pada tabel SUPPLIER.

Skema barunya adalah

SUPPLIER_DETAIL = (SID, SADDRESS, SCITY, RATING)
SUPPLIER_ MAIN = (SID, SNAME, BALANCE)
FK: SUPPLIER_DETAIL (SID) → SUPPLIER_MAIN(SID)

Karena ada foreign key reference dari tabel SUPPLIER_DETAIL ke SUPPLIER_MAJN, SID pada tabel SUPPLIER_DETAIL merupakan salah satu SID pada SUPPLIER_MAIN. Sebaliknya, harus dipastikan bahwa semua SID yang muncul pada tabel SUPPLIER_MAIN juga muncul di tabel SUPPLIER_DETAIL karena merupakan hasil splitting.

Kasus 4

Untuk mengatasi permasalahan “Pencarian data supplier dan product sangat sering dilakukan berdasarkan nama.”, perlu dilakukan **index tuning.**

Relasi SUPPLIER_MAIN, atribut SNAME.

Jenis indeks yang bisa dipilih ada 2, yaitu:

1. **Hash-index**, index ini baik untuk operasi lookup yang sederhana dan cepat berdasarkan kondisi kesamaan.
2. **B+tree**, secara rata-rata, dibutuhkan waktu yang sama untuk mengakses baris manapun dalam index sehingga menjadi jenis index yang paling umum untuk diimplementasikan di DBMS. Index ini digunakan pada atribut dengan high data sparsity alias data dengan nilai berulang cukup sedikit (seperti data nama).

Kasus 5

Pada kasus ini terdapat 2 permasalahan,

1. Untuk mengatasi permasalahan “Untuk memenuhi kebutuhan dari pihak eksekutif, query untuk mendapatkan total harga terhutang dari seluruh pesanan yang belum diterima, baik **per *supplier*** sering dijalankan”, perlu dibuat **materialized view**.

   Skema materialized view-nya adalah

   ```
   CREATE VIEW VHARGAPERSID AS
   SELECT SID, SUM(QUANTITY * PRICE) AS TOTPRICE
   ```

FROM SUPPLY_NOTRECEIVED
GROUP BY SID

2. Untuk mengatasi permasalahan “Untuk memenuhi kebutuhan dari pihak eksekutif, query untuk mendapatkan total harga terhutang dari seluruh pesanan yang belum diterima, baik **per tanggal** sering dijalankan”, perlu dibuat **materialized view**.
   Skema materialized view-nya adalah

   CREATE VIEW VHARGAPERORDERDATE AS
   SELECT ORDERDATE, SUM(QUANTITY * PRICE) AS TOTPRICE
   FROM SUPPLY_NOTRECEIVED
   GROUP BY ORDERDATE

Final Schema

PRODUCT = (PID, PNAME, CID, CNAME, WEIGHT)
SUPPLY_RECEIVED = (ORDERDATE, ***PID, SID***, QUANTITY, PRICE, RECEIVEDATE)
SUPPLY_NOTRECEIVED = (ORDERDATE, ***PID, SID***, QUANTITY, PRICE, RECEIVEDATE)
SUPPLIER_DETAIL = (*SID*, SADDRESS, SCITY, RATING)
SUPPLIER_ MAIN = (*SID*, SNAME, BALANCE)

Skema materialized view 1

CREATE VIEW VHARGAPERSID AS
SELECT SID, SUM(QUANTITY * PRICE) AS TOTPRICE
FROM SUPPLY_NOTRECEIVED
GROUP BY SID

Skema materialized view 2

CREATE VIEW VHARGAPERORDERDATE AS
SELECT ORDERDATE, SUM(QUANTITY * PRICE) AS TOTPRICE
FROM SUPPLY_NOTRECEIVED
GROUP BY ORDERDATE