

**LAPORAN WORKSHOP
BASIS DATA LANJUT**

Politeknik Caltex Riau

LATIHAN SOAL PENGAYAAN

Dosen/PLP :

Kartina Diah Kusuma Wardhani, S.T., M.T.

Asmarini, S.Tr.Kom

Identitas :

Nama Lengkap : Davin Williem

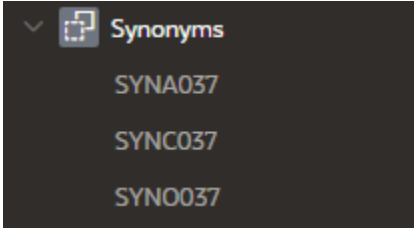
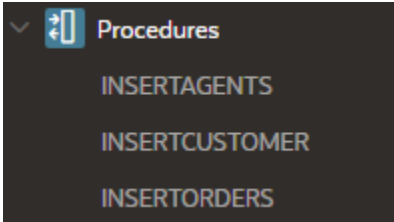
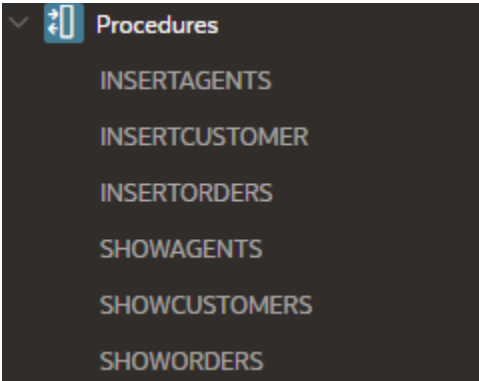
NIM : 2255301037

Kelas : 1 TI A

**PRODI TEKNIK INFORMATIKA
JURUSAN TEKNOLOGI INFORMASI**

LATIHAN SOAL PENGAYAAN

A. Latihan Praktikum

No	Perintah SQL	Hasil	Analisa
1	<pre>--1 CREATE SYNONYM synA037 FOR Agents; CREATE SYNONYM synC037 FOR Customer; CREATE SYNONYM synO037 FOR Orders;</pre>	<p>Synonym created.</p> 	Membuat SYNONYM synA037 untuk tabel Agents, SYNONYM synC037 untuk tabel Customer dan SYNONYM synO037 untuk tabel Orders.
2	<pre>--2 CREATE OR REPLACE PROCEDURE insertAgents (ac CHAR(3), an VARCHAR2(30), wa VARCHAR2(30), co VARCHAR2(30), c VARCHAR2(30), ph VARCHAR2(30)) AS BEGIN INSERT INTO Agents (agent_code, agent_name, working_area, commission, phone_no, country) VALUES (ac, an, wa, c, ph, co); END; CREATE OR REPLACE PROCEDURE insertCustomer (cu CHAR(3), cn VARCHAR2(30), ca VARCHAR2(30), co VARCHAR2(30), c VARCHAR2(30), ph VARCHAR2(30)) AS BEGIN INSERT INTO Customer (customer_code, customer_name, customer_address, commission, phone_no, country) VALUES (cu, cn, ca, c, ph, co); END; CREATE OR REPLACE PROCEDURE insertOrders (o CHAR(3), oc VARCHAR2(30), oa VARCHAR2(30), co VARCHAR2(30), c VARCHAR2(30), ph VARCHAR2(30)) AS BEGIN INSERT INTO Orders (order_code, order_customer, order_agent, commission, phone_no, country) VALUES (o, oc, oa, c, ph, co); END;</pre>	<p>Procedure created.</p> 	Membuat PROCEDURE insertAgents untuk memasukkan data ke dalam tabel Agents dengan parameter, PROCEDURE insertCustomer untuk memasukkan data ke dalam tabel Customer dengan parameter dan PROCEDURE insertOrders untuk memasukkan data ke dalam tabel Orders dengan parameter.
3	<pre>CREATE OR REPLACE PROCEDURE showAgents (ac CHAR) IS dataAgents agents%ROWTYPE; BEGIN SELECT * INTO dataAgents FROM Agents WHERE Agent_Code = ac; DBMS_OUTPUT.PUT_LINE(dataAgents.agent_code ',' dataAgents.Agent_Name ',' dataAgents.Working_Area ',' dataAgents.Commission ',' dataAgents.phone_no ',' dataAgents.Country); END;</pre>	<p>Procedure created.</p> 	Membuat PROCEDURE showAgents untuk menampilkan data yang ada pada tabel Agents dengan menggunakan dataAgents yang diambil dari setiap baris yang ada pada tabel Agents, PROCEDURE showCustomers untuk menampilkan data yang ada pada tabel Customer dengan menggunakan dataCust yang diambil dari

	<pre> CREATE OR REPLACE PROCEDURE showCustomers (cc CHAR) IS dataCust Customers%ROWTYPE; BEGIN SELECT * INTO dataCust FROM Customers WHERE Cust_Code = cc; DBMS_OUTPUT.PUT_LINE(dataCust.Cust_Code ',' dataCust.Cust_Name ',' dataCust.Cust_City ',' dataCust.Working_Area ',' dataCust.Cust_Country ',' dataCust.Grade ',' dataCust.Opening_AMT ',' dataCust.Receive_AMT ',' dataCust.Payment_AMT ',' dataCust.Outstanding_AMT ',' dataCust.phone_no ',' dataCust.Agent_Code); END; CREATE OR REPLACE PROCEDURE showOrders (orn CHAR) IS dataOrd Orders%ROWTYPE; BEGIN SELECT * INTO dataOrd FROM Orders WHERE Ord_Num= orn; DBMS_OUTPUT.PUT_LINE(dataOrd.Ord_Num ',' dataOrd.Ord_Amount ',' dataOrd.Advance_Amount ',' dataOrd.Ord_Date ',' dataOrd.Cust_Code ',' dataOrd.Agent_Code ',' dataOrd.Ord_Description); END; </pre>	<p>setiap baris yang ada pada tabel Customer dan PROCEDURE showOrders untuk menampilkan data yang ada pada tabel Orders dengan menggunakan dataOrd yang diambil dari setiap baris yang ada pada tabel Orders.</p>
4	<pre> CREATE VIEW transaksi AS SELECT Cust_Code, Agent_Code, Ord_Date, Ord_Amount FROM Orders; CREATE SYNONYM tampilkanTransaksi FOR transaksi; </pre>	 <p>Membuat VIEW transaksi untuk menampilkan Cust_Code, Agent_Code, Ord_Date, Ord_Amount dari tabel Orders dan membuat SYNONYM tampilkanTransaksi untuk VIEW transaksi.</p>

5

```

CREATE OR REPLACE PROCEDURE
displayData (tableName VARCHAR2, idData CHAR) IS
    dataA Agents%ROWTYPE;
    dataC Customer%ROWTYPE;
    dataO_Ord_Date Orders.ord_date%TYPE;
    dataO_Agent_Name Agents.Agent_Name%TYPE;
    dataO_Cust_Name Customer.Cust_Name%TYPE;
BEGIN
    CASE UPPER(tableName)
        WHEN 'AGENTS' THEN
            SELECT
                *
            INTO
                dataA
            FROM
                Agents
            WHERE
                UPPER(Agent_Code) = UPPER(idData);
        WHEN 'CUSTOMER' THEN
            SELECT
                *
            INTO
                dataC
            FROM
                Customer
            WHERE
                UPPER(Cust_Code) = UPPER(idData);
        DBMS_OUTPUT.PUT_LINE('Nama Customer: ' || dataC.Cust_Name);
    WHEN 'ORDERS' THEN
        SELECT
            Agent_Name,
            Cust_Name,
            Ord_Date
        INTO

```

```

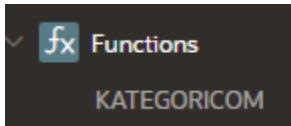
        dataO_Agent_Name,
        dataO_Cust_Name,
        dataO_Ord_Date
    FROM
        Orders o,
        Agents a,
        Customer c
    WHERE
        UPPER(Ord_Num) = UPPER(idData)
    AND
        o.Cust_Code = c.Cust_Code
    AND
        o.Agent_Code = a.Agent_Code;
    DBMS_OUTPUT.PUT_LINE('Nama Agent: ' || dataO_Agent_Name);
    DBMS_OUTPUT.PUT_LINE('Nama Customer: ' || dataO_Cust_Name);
    DBMS_OUTPUT.PUT_LINE('Tanggal: ' || dataO_Ord_Date);
    ELSE
        DBMS_OUTPUT.PUT_LINE('Tabel Tidak Adal!');
    END CASE;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Data Tidak Adal!');
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Error!');
END;
```

Procedure created.

Procedures

- DISPLAYDATA
- INSERTAGENTS
- INSERTCUSTOMER
- INSERTORDERS
- SHOWAGENTS
- SHOWCUSTOMERS
- SHOWORDERS

Membuat PROCEDURE displayData yang menggunakan dataA yang diambil dari setiap baris yang ada pada tabel Agents, dataC yang diambil dari setiap baris yang ada pada tabel Customer, dataO_Ord_Date yang diambil dari Ord_Date yang ada pada tabel Orders, dataO_Agent_Name yang diambil dari Agent_Name yang ada pada tabel Agents dan dataO_Cust_Name yang diambil dari Cust_Name yang ada pada tabel Customer. Lalu di dalam kode BEGIN yang berisi CASE UPPER(tableName) dimana ketika 'AGENTS', maka menampilkan semua data ke dalam dataA dari tabel Agents dimana UPPER dari Agent_Code sama dengan UPPER dari idData. Jika 'CUSTOMER', maka menampilkan semua data ke dalam dataC dari tabel Customer dimana UPPER dari Cust_Code sama dengan UPPER dari idData dan menampilkan Nama Customer. Jika 'ORDERS', maka menampilkan Agent_Name, Cust_Name dan Ord_Date ke dalam dataO_Agent_Name, dataO_Cust_Name, dan dataO_Ord_Date dari tabel Orders, Agents, dan Customer dimana UPPER dari Ord_Num sama dengan UPPER dari idData dan Cust_Code dari

			<p>tabel Orders sama dengan Cust_Code dari tabel Customer dan Agent_Code dari tabel Orders sama dengan Agent_Code dari tabel Agents lalu menampilkan Nama Agent, Nama Customer dan Tanggal. Jika tidak ada tableName yang ditemukan, maka menampilkan kata "Tabel Tidak Ada!". Ketika data tidak ditemukan, maka menampilkan kata "Data Tidak Ada!" dan ketika muncul selain data tidak ada, maka menampilkan kata "Error!".</p>
6	<pre>--6 CREATE FUNCTION kategoriCom (comm IN NUMBER) RETURN CHAR IS BEGIN IF(comm < 0.25) THEN RETURN 'C'; ELSIF (comm >= 0.25 AND comm <= 0.5) THEN RETURN 'B'; ELSE RETURN 'A'; END IF; END;</pre>	<p>Function created.</p> 	<p>Membuat FUNCTION kategoriCom dengan parameter comm dalam NUMBER dan mengembalikan CHAR jika comm lebih kecil dari 0.25 maka mengembalikan CHAR 'C'. Jika comm lebih besar sama dengan 0.25 dan comm lebih kecil sama dengan 0.5, maka mengembalikan CHAR 'B' dan jika tidak maka mengembalikan char 'A'.</p>
7	<pre>CREATE OR REPLACE PROCEDURE tambahKomisi(nominal NUMBER) IS tmp NUMBER(10, 2) := nominal; tmp NUMBER(10); avgCom Agents.commissionTYPE; dataAgents AgentsnomTYPE; CURSOR cAgents IS SELECT * FROM Agents; BEGIN SELECT AVG(commission) INTO avgCom FROM Agents; OPEN cAgents; LOOP FETCH cAgents INTO dataAgents; EXIT WHEN cAgents%NOTFOUND; IF(dataAgents.commission < avgCom) AND (tmp >= 0) THEN tmp := (nominal * dataAgents.commission / avgCom); IF(tmp >= tmp) THEN tmp := tmp; ELSE tmp := tmp - tmp; END IF; END IF; UPDATE Agents SET commission = commission + tmp WHERE agent_code = dataAgents.agent_code; END LOOP; CLOSE cAgents; END;</pre> <p>BEGIN tambahKomisi(5); END;</p>	<p>Procedure created.</p>   <p>Statement processed.</p>	<p>Membuat PROCEDURE tambahKomisi dengan parameter nominal NUMBER. Lalu mendeklarasikan variabel tmp dengan tipe data NUMBER yang bernilai variabel nominal dan variabel tmb dengan tipe data NUMBER. Lalu ada avgCom yang diambil dari Commission dari tabel Agents, dataAgents yang diambil dari</p>

AGENT_CODE	AGENT_NAME	REMARKS	DATE	COMMISSION	FIXED_FEE	CURSOR
001	John Doe	Agent	1/1/2020	100	1000000	001
002	John Doe	Agent	1/1/2020	100	1000000	002

setiap baris yang ada pada tabel Agents dan CURSOR cAgents yang menampilkan semua data dari tabel Agents. Di dalam kode BEGIN, terdapat query SELECT untuk menampilkan rata-rata dari Commission ke dalam avgCom dari tabel Agents. Lalu membuka cAgents dan LOOP untuk perulangan. FETCH cAgents ke dalam dataAgents yang digunakan untuk mengambil data dari cAgents dan akan keluar dari perulangan ketika cAgents tidak ditemukan lagi. Jika Commission pada dataAgents lebih kecil dari avgCom dan tmp lebih besar sama dengan 0, maka tmb adalah nominal dikali Commission dari dataAgents dibagi dengan rata-rata dari Commission. Jika tmb lebih besar dari tmp, maka tmb adalah tmp dan tmp sama dengan 0. Jika tidak keduanya, maka tmp adalah tmp dikurang dengan tmb. Mengubah Commission pada tabel Agents dengan Commission ditambah dengan tmb dimana Agent_Code sama dengan Agent_Code yang ada pada dataAgents. Lalu menutup cAgents. Setelah itu, menggunakan PROCEDURE tambahKomisi dengan nominal 5 di dalam parameter.

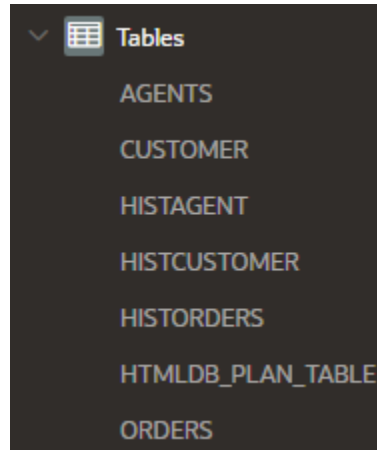
8

```
CREATE TABLE histCustomer (
  idhistC NUMBER PRIMARY KEY,
  tanggal timestamp,
  Cust_Code VARCHAR2(6),
  Cust_Name VARCHAR2(40),
  Cust_City CHAR(35),
  Working_Area VARCHAR2(35),
  Cust_Country VARCHAR2(20),
  Grade NUMBER,
  Opening_AMT NUMBER,
  Receive_AMT NUMBER,
  Payment_AMT NUMBER,
  Outstanding_AMT NUMBER,
  Phone_No VARCHAR2(17),
  Agent_Code CHAR(6)
);
```

```
CREATE TABLE histOrders (
  idhistO NUMBER,
  tanggal timestamp,
  Order_Num DECIMAL(6,0),
  Ord_Amount DECIMAL(12,2),
  Advance_Amount DECIMAL(12,2),
  Ord_Date date,
  Cust_Code VARCHAR(6),
  Agent_Code CHAR(6),
  Ord_Description VARCHAR(60),
  CONSTRAINT pk_histOrder PRIMARY KEY(idhistO)
);
```

```
CREATE TABLE histAgent (
  idHistA NUMBER PRIMARY KEY,
  tanggal timestamp,
  Agent_Name CHAR(40),
  Working_Area CHAR(35),
  Commission NUMBER(10,2),
  Phone_No CHAR(15),
  Country VARCHAR(25)
);
```

Table created.

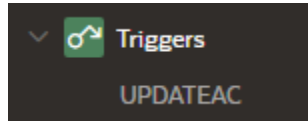


Membuat tabel histCustomer dengan kolom idhistC sebagai primary key, tanggal, Cust_Code, Cust_Name, Cust_City, Working_Area, Cust_Country, Grade, Opening_AMT, Receive_AMT, Payment_AMT, Outstanding_AMT, Phone_No dan Agent_Code. Membuat tabel histOrders dengan kolom idhistO, tanggal, Order_Num, Ord_Amount, Advance_Amount, Ord_Date, Cust_Code, Agent_Code, Ord_Description dan membuat CONSTRAINT pk_histOrder sebagai primary key untuk kolom idhistO. Membuat tabel histAgent dengan kolom idHistA sebagai primary key, tanggal, Agent_Name, Working_Area, Commission, Phone_No dan Country.

9

```
--9
CREATE OR REPLACE TRIGGER updateAC
BEFORE UPDATE ON Agents
FOR EACH ROW
BEGIN
    UPDATE
    Customer
    SET
    Agent_Code = :NEW.Agent_Code
    WHERE
    Agent_Code = :OLD.Agent_Code;
    UPDATE
    Orders
    SET
    Agent_Code = :NEW.Agent_Code
    WHERE
    Agent_Code = :OLD.Agent_Code;
END;
```

Trigger created.



Membuat TRIGGER updateAC yang mengeksekusi kode di dalam BEGIN untuk mengubah data pada tabel Customer dari Agent_Code lama menjadi Agent_Code yang baru dan mengubah data pada tabel Orders dari Agent_Code lama menjadi Agent_Code yang baru