



**LAPORAN PRAKTIKUM  
Basis Data Lanjut**

**Pengampu :**

**Muhammad Naufal Luthfi, S.T., M.Kom  
Rian Ardi Wibowo, S.Tr.Kom**

**Identitas :**

No Absen : 4

Nama Lengkap : Arini Zahira Putri

NIM : 2457301017

Kelas : 2 SI B

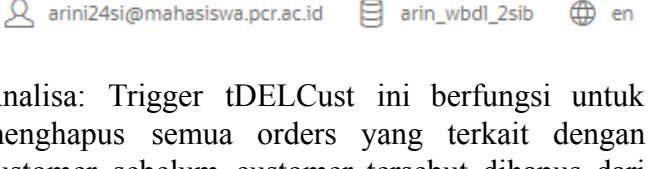
**Judul Praktikum : PLSQLExceptionTrigger**

**Hari/Tanggal : Rabu / 03 Desember 2025**

## MODUL XII EXCEPTION - TRIGGER

### LATIHAN PRAKTIKUM PLSQLExceptionTrigger

NO	QUERY	ANALISA
1	<p>Instruksi Tambahan: Jelaskan tujuan dari perintah di samping! Jika terjadi error, identifikasilah penyebab terjadinya error dan lakukan perbaikan agar berjalan dengan benar!</p> <p>Before:</p> <pre>CREATE OR REPLACE TRIGGER tShowDataAgents BEFORE INSERT ON agents FOR EACH ROW BEGIN dbms_output.put_line('Agent Code: '    :NEW.AGENT_CODE); dbms_output.put_line('Agent Name: '    :NEW.AGENT_NAME); dbms_output.put_line('COMISSION : '    :NEW.COMISSION); END;</pre> <p>After penulisan 'COMISSION' seharusnya 'COMMISSION':</p> <pre>CREATE OR REPLACE TRIGGER tShowDataAgents BEFORE INSERT ON agents FOR EACH ROW BEGIN dbms_output.put_line('Agent Code: '    :NEW.AGENT_CODE); dbms_output.put_line('Agent Name: '    :NEW.AGENT_NAME); dbms_output.put_line('COMMISSION : '    :NEW.COMMISSION); END;</pre>	<p>Error at line 4: PLS-00049: bad bind variable 'NEW.COMISSION'</p> <pre>2. BEFORE INSERT ON agents 3. FOR EACH ROW 4. BEGIN 5. dbms_output.put_line('Agent Code: '    :NEW.AGENT_CODE); 6. dbms_output.put_line('Agent Name: '    :NEW.AGENT_NAME);</pre> <p>Trigger created.</p> <p>0.07 seconds</p> <p>Profile: arini24si@mahasiswa.pcr.ac.id    Database: arin_wbdl_2sib    Language: en</p> <p>Analisa: Trigger ini bertujuan menampilkan data agent yang baru diinsert sebelum data tersebut disimpan ke tabel agents. Terjadi error karena penulisan kolom 'COMISSION' salah, seharusnya 'COMMISSION' sesuai nama kolom di tabel agents. Setelah diperbaiki, trigger akan menampilkan Agent Code, Agent Name, dan Commission setiap kali ada insert data baru.</p>

2	<p>Instruksi Tambahan: Insertkan sebuah data pada table customer dan lihat hasilnya!</p> <pre>CREATE OR REPLACE TRIGGER tINSCust AFTER INSERT ON customer FOR EACH ROW BEGIN DBMS_OUTPUT.PUT_LINE('Dari Trigger tINSCust'); DBMS_OUTPUT.PUT_LINE('Cust Code :'  :NEW.CUST_CODE); DBMS_OUTPUT.PUT_LINE('Cust NAME :'  :NEW.CUST_NAME); END;</pre> <p>Insert data customer untuk testing trigger:</p> <pre>INSERT INTO customer VALUES ('C99999', 'Arini Zahira Putri', 'Pekanbaru', 'Pekanbaru', 'Indonesia', 2, 8000, 5000, 3000, 10000, '081234567890', 'A014');</pre> <p>Lihat data/tabel yg dibuat:</p> <pre>SELECT * FROM customer WHERE CUST_CODE = 'C99999';</pre>	<p><b>Trigger created.</b></p> <p><b>0.07 seconds</b></p>  <p>Dari Trigger tINSCust Cust Code : C99999 Cust NAME : Arini Zahira Putri 1 row(s) inserted.</p> <p><b>Customer Table Data:</b></p> <table border="1"> <thead> <tr> <th>CUST_CODE</th> <th>CUST_NAME</th> <th>CUST_CITY</th> <th>WORKING_AREA</th> <th>COUNTRY_CODE</th> <th>GRADE</th> <th>OPENING_AMT</th> <th>SECCES_AMT</th> <th>PAYOUT_AMT</th> <th>OUTSTANDING_AMT</th> <th>PHONE_NO</th> <th>ASSET_CODE</th> </tr> </thead> <tbody> <tr> <td>C99999</td> <td>Arini Zahira Putri</td> <td>Pekanbaru</td> <td>Pekanbaru</td> <td>Indonesia</td> <td>2</td> <td>8000</td> <td>5000</td> <td>3000</td> <td>10000</td> <td>081234567890</td> <td>A014</td> </tr> </tbody> </table> <p>Analisa: Trigger tINSCust berfungsi menampilkan informasi customer yang baru diinsert setelah proses insert berhasil. Ketika data customer dengan kode 'C99999' atas nama 'Arini Zahira Putri' diinsert, trigger otomatis menampilkan pesan "Dari Trigger tINSCust" beserta Cust Code dan Cust Name, lalu data berhasil tersimpan di tabel customer yang dapat dilihat dengan query SELECT.</p>	CUST_CODE	CUST_NAME	CUST_CITY	WORKING_AREA	COUNTRY_CODE	GRADE	OPENING_AMT	SECCES_AMT	PAYOUT_AMT	OUTSTANDING_AMT	PHONE_NO	ASSET_CODE	C99999	Arini Zahira Putri	Pekanbaru	Pekanbaru	Indonesia	2	8000	5000	3000	10000	081234567890	A014
CUST_CODE	CUST_NAME	CUST_CITY	WORKING_AREA	COUNTRY_CODE	GRADE	OPENING_AMT	SECCES_AMT	PAYOUT_AMT	OUTSTANDING_AMT	PHONE_NO	ASSET_CODE															
C99999	Arini Zahira Putri	Pekanbaru	Pekanbaru	Indonesia	2	8000	5000	3000	10000	081234567890	A014															
3	<pre>CREATE OR REPLACE TRIGGER tDEL Cust BEFORE DELETE ON customer FOR EACH ROW BEGIN DELETE FROM ORDERS WHERE CUST_CODE = :OLD.CUST_CODE; END;</pre>	<p><b>Trigger created.</b></p> <p><b>0.07 seconds</b></p>  <p>Analisa: Trigger tDEL Cust ini berfungsi untuk menghapus semua orders yang terkait dengan customer sebelum customer tersebut dihapus dari tabel, sehingga tidak terjadi error foreign key constraint karena masih ada data orders yang mereferensi customer yang akan dihapus (cascade delete manual).</p>																								

4	<p>Instruksi Tambahan: Jika terjadi error, identifikasilah penyebab terjadinya error dan lakukan perbaikan agar berjalan dengan benar!</p> <pre>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) );</pre> <p>After:</p> <pre>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(50), grade NUMBER, opening_amt NUMBER, receive_amt NUMBER, payment_amt NUMBER, outstanding_amt NUMBER, phone_no VARCHAR2(17), agent_code CHAR(6) );</pre>	 <p>The screenshot shows the Oracle SQL Developer interface. A query window contains the SQL command to create the histCustomer table. The output pane shows the results of the execution: "Table created." and "0.06 seconds". Below the output, there is user information: arini24si@mahasiswa.pcr.ac.id, arin_wbdl_2sib, and en. The bottom section of the screenshot contains an analysis of the query.</p> <p><b>Analisa:</b> Query ini membuat tabel histCustomer untuk menyimpan backup data customer yang dihapus. Error terjadi karena tipe data cust_country VARCHAR2(20) terlalu kecil untuk menampung nama negara yang panjang, sehingga diperbaiki menjadi VARCHAR2(50) agar dapat menampung data negara dengan nama yang lebih panjang.</p>
---	--	---

5	<pre>CREATE SEQUENCE idHisDelCu_seq START WITH 1;</pre>	<p>Sequence created.</p> <p>0.04 seconds</p> <hr/> <p> arini24si@mahasiswa.pcr.ac.id  arin_wbdl_2sib  en</p> <p>Analisa: Query ini membuat sequence idHisDelCu_seq yang dimulai dari angka 1, yang akan digunakan sebagai auto-increment untuk kolom idhistC (primary key) pada tabel histCustomer setiap kali ada data customer yang dibackup.</p>
6	<pre>CREATE OR REPLACE TRIGGER tDEL Cust BEFORE DELETE ON customer FOR EACH ROW BEGIN INSERT into histCustomer values( idHisDelCu_seq.NEXTVAL, localtimestamp, :OLD.CUST_CODE, :OLD.CUST_NAME, :OLD.CUST_CITY, :OLD.WORKING_AREA, :OLD.CUST_COUNTRY, :OLD.GRADE, :OLD.OPENING_AMT, :OLD.RECEIVE_AMT, :OLD.PAYMENT_AMT, :OLD.OUTSTANDING_AMT, :OLD.PHONE_NO, :OLD.AGENT_CODE ); END;</pre>	<p>Trigger created.</p> <p>0.08 seconds</p> <hr/> <p> arini24si@mahasiswa.pcr.ac.id  arin_wbdl_2sib  en</p> <p>Analisa: Trigger tDEL Cust ini diperbarui untuk membackup data customer yang akan dihapus ke tabel histCustomer menggunakan sequence idHisDelCu_seq untuk generate ID otomatis, serta menyimpan timestamp dan semua field dari customer lama menggunakan :OLD sebelum data dihapus.</p>

7	<p>Instruksi Tambahan: Analisis hasil yang terjadi pada table customer dan histCustomer!</p> <pre>DELETE FROM CUSTOMER WHERE CUST_CODE = 'C00100';</pre>	<p><code>0 row(s) deleted.</code></p> <p><code>0.03 seconds</code></p> <p> arini24si@mahasiswa.pcr.ac.id  arin_wbdl_2sib  en</p> <p>Analisa: Ketika customer dengan kode 'C00100' dihapus, trigger tDELCust otomatis memindahkan semua data customer tersebut ke tabel histCustomer sebagai backup, sehingga data customer hilang dari tabel customer namun tersimpan di histCustomer dengan timestamp penghapusan, dan jika customer memiliki orders maka orders tersebut juga ikut terhapus.</p>
8	<p>Instruksi Tambahan: Jika terjadi error, identifikasi penyebab terjadinya error dan lakukan perbaikan agar berjalan dengan benar ! Jelaskan perbedaan antara query nomor 4 dan nomor 8!</p> <pre>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) );</pre>	<p><code>Table created.</code></p> <p><code>0.06 seconds</code></p> <p> arini24si@mahasiswa.pcr.ac.id  arin_wbdl_2sib  en</p> <p>Analisa: Query ini membuat tabel histOrders untuk menyimpan backup orders yang dihapus. Error terjadi karena penggunaan VARCHAR tanpa angka 2 (seharusnya VARCHAR2) untuk kolom cust_code dan ord_description di Oracle SQL. Perbedaan dengan nomor 4 adalah nomor 4 membuat tabel backup untuk customer, sedangkan nomor 8 membuat tabel backup untuk orders dengan struktur kolom yang berbeda sesuai tabel orders.</p>

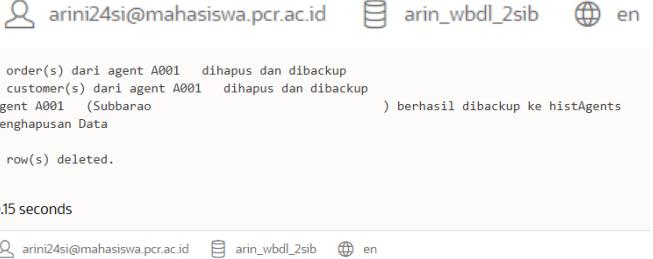
9	<pre>CREATE SEQUENCE idHisDelOr_seq START WITH 1;</pre>	<p>Sequence created.</p> <p>0.06 seconds</p> <p> arini24si@mahasiswa.pcr.ac.id  arin_wbdl_2sib  en</p> <p>Analisa: Query ini membuat sequence idHisDelOr_seq dimulai dari 1 yang akan digunakan untuk generate primary key otomatis pada tabel histOrders setiap kali ada data orders yang dibackup saat dihapus.</p>
10	<pre>CREATE OR REPLACE TRIGGER tDELOrder BEFORE DELETE ON orders FOR EACH ROW BEGIN INSERT into histOrders values ( idHisDelOr_seq.NEXTVAL, localtimestamp, :OLD.ORD_NUM, :OLD.ORD_AMOUNT, :OLD.ADVANCE_AMOUNT, :OLD.ORD_DATE, :OLD.CUST_CODE, :OLD.AGENT_CODE, :OLD.ORD_DESCRIPTION ); END;</pre>	<p>Trigger created.</p> <p>0.08 seconds</p> <p> arini24si@mahasiswa.pcr.ac.id  arin_wbdl_2sib  en</p> <p>Analisa: Trigger tDELOrder berfungsi membackup data orders yang akan dihapus ke tabel histOrders sebelum proses delete, dengan menyimpan semua informasi order menggunakan :OLD dan ID otomatis dari sequence idHisDelOr_seq beserta timestamp penghapusan.</p>
11	<p>Instruksi Tambahan: Analisis hasil yang terjadi pada table orders dan histOrders!</p> <pre>DELETE FROM orders WHERE ORD_NUM='200100';</pre>	<p>1 row(s) deleted.</p> <p>0.07 seconds</p> <p> arini24si@mahasiswa.pcr.ac.id  arin_wbdl_2sib  en</p> <p>Analisa: Ketika order dengan ORD_NUM='200100' dihapus, trigger tDELOrder</p>

		otomatis memindahkan data order tersebut ke tabel histOrders sebagai backup sebelum dihapus dari tabel orders, sehingga data order tidak hilang permanen dan dapat dilihat di tabel histOrders dengan timestamp kapan data tersebut dihapus.
12	<pre> CREATE OR REPLACE TRIGGER tDELCust BEFORE DELETE ON customer FOR EACH ROW BEGIN INSERT into histCustomer values ( idHisDelCu_seq.NEXTVAL, localtimestamp, :OLD.CUST_CODE, :OLD.CUST_NAME, :OLD.CUST_CITY, :OLD.WORKING_AREA, :OLD.CUST_COUNTRY, :OLD.GRADE, :OLD.OPENING_AMT, :OLD.RECEIVE_AMT, :OLD.PAYMENT_AMT, :OLD.OUTSTANDING_AMT, :OLD.PHONE_NO, :OLD.AGENT_CODE ); DELETE FROM orders WHERE CUST_CODE = :OLD.CUST_CODE; END; </pre>	<p>Trigger created. 0.07 seconds</p> <p>arini24si@mahasiswa.pcr.ac.id arin_wbdl_2sib en</p> <p>Analisa: Trigger tDELCust ini diperbarui dengan menggabungkan fungsi backup dan cascade delete, sehingga ketika customer dihapus, data customer dibackup ke histCustomer terlebih dahulu, lalu semua orders yang terkait dengan customer tersebut juga dihapus (yang otomatis akan dibackup ke histOrders oleh trigger tDELOrder).</p>
13	<p>Instruksi Tambahan: Analisis hasil yang terjadi pada table customer, orders, histCustomer dan histOrders !</p> <pre> DELETE FROM customer WHERE CUST_CODE='C00007'; </pre>	<p>1 row(s) deleted. 0.10 seconds</p> <p>arini24si@mahasiswa.pcr.ac.id arin_wbdl_2sib en</p> <p>Analisa: Ketika customer 'C00007' dihapus, trigger tDELCust membackup data customer ke histCustomer, kemudian menghapus semua orders</p>

		terkait customer tersebut yang akan memicu trigger tDELOrder untuk membackup orders ke histOrders, sehingga terjadi cascade backup dimana data customer dan semua orders-nya tersimpan di tabel history masing-masing sebelum dihapus dari tabel utama.
14	<p>Instruksi Tambahan: Analisis syntax nomor 14 dan buktikan dengan hasil tangkapan layar!</p> <pre>CREATE OR REPLACE TRIGGER t BEFORE INSERT OR UPDATE OF AGENT_CODE, AGENT_NAME OR DELETE ON AGENTS FOR EACH ROW BEGIN CASE WHEN INSERTING THEN DBMS_OUTPUT.PUT_LINE('Input Data Pada Tabel AGENTS'); WHEN UPDATING('AGENT_CODE') THEN DBMS_OUTPUT.PUT_LINE('Perubahan Data Pada Kolom AGENT_CODE dari Tabel AGENTS'); WHEN UPDATING('AGENT_NAME') THEN DBMS_OUTPUT.PUT_LINE('Perubahan Data Pada Kolom AGENT_NAME : '  :OLD.AGENT_NAME   ' menjadi : '  :NEW.AGENT_NAME); WHEN DELETING THEN DBMS_OUTPUT.PUT_LINE('Penghapusan Data'); END CASE; END;</pre>	<p>Trigger created.</p> <p>0.07 seconds</p> <p> arini24si@mahasiswa.pcr.ac.id  arin_wbdl_2sib  en</p> <p>Analisa: Trigger t adalah compound trigger yang dapat mendeteksi 4 jenis operasi DML (INSERT, UPDATE pada AGENT_CODE atau AGENT_NAME, dan DELETE) pada tabel AGENTS dalam satu trigger. Menggunakan CASE statement untuk menampilkan pesan berbeda sesuai operasi yang dilakukan: pesan "Input Data" saat insert, "Perubahan Data Pada Kolom AGENT_CODE" saat update agent_code, menampilkan perubahan nama lama ke nama baru saat update agent_name, dan "Penghapusan Data" saat delete, sehingga satu trigger dapat memonitor berbagai perubahan data.</p>

## TUGAS

NO	QUERY	ANALISA																																																																																				
1	<p>Buatlah sebuah table untuk menampung data yang dihapus dari table agents!</p> <pre>CREATE TABLE histAgents( idhistA NUMBER PRIMARY KEY, tanggal TIMESTAMP, agent_code CHAR(6), agent_name VARCHAR2(40), working_area CHAR(35), commission NUMBER(12,2), phone_no CHAR(15), country VARCHAR2(25) );  Sequence: CREATE SEQUENCE idHisDelAg_seq START WITH 1;  DESC histAgents;  SELECT table_name FROM user_tables WHERE table_name = 'HISTAGENTS';</pre>	<p>Table created.</p> <p>0.08 seconds</p> <p>Sequence created.</p> <p>0.04 seconds</p> <table border="1"> <thead> <tr> <th>Object Type</th> <th>TABLE</th> <th>Object</th> <th>HISTAGENTS</th> </tr> </thead> <tbody> <tr> <td>Table</td> <td>IDHISTA</td> <td>NUMBER</td> <td>22</td> <td>-</td> <td>-</td> <td>1</td> <td>-</td> <td>-</td> <td>-</td> </tr> <tr> <td></td> <td>TANGGAL</td> <td>TIMESTAMP(6)</td> <td>11</td> <td>-</td> <td>6</td> <td>-</td> <td>✓</td> <td>-</td> <td>-</td> </tr> <tr> <td></td> <td>AGENT_CODE</td> <td>CHAR</td> <td>6</td> <td>-</td> <td>-</td> <td>-</td> <td>✓</td> <td>-</td> <td>-</td> </tr> <tr> <td></td> <td>AGENT_NAME</td> <td>VARCHAR2</td> <td>40</td> <td>-</td> <td>-</td> <td>-</td> <td>✓</td> <td>-</td> <td>-</td> </tr> <tr> <td></td> <td>WORKING_AREA</td> <td>CHAR</td> <td>35</td> <td>-</td> <td>-</td> <td>-</td> <td>✓</td> <td>-</td> <td>-</td> </tr> <tr> <td></td> <td>COMMISSION</td> <td>NUMBER</td> <td>-</td> <td>12</td> <td>2</td> <td>-</td> <td>✓</td> <td>-</td> <td>-</td> </tr> <tr> <td></td> <td>PHONE_NO</td> <td>CHAR</td> <td>15</td> <td>-</td> <td>-</td> <td>-</td> <td>✓</td> <td>-</td> <td>-</td> </tr> <tr> <td></td> <td>COUNTRY</td> <td>VARCHAR2</td> <td>25</td> <td>-</td> <td>-</td> <td>-</td> <td>✓</td> <td>-</td> <td>-</td> </tr> </tbody> </table> <p>TABLE_NAME</p> <p>HISTAGENTS</p> <p>1 rows returned in 0.06 seconds <a href="#">Download</a></p> <p>arin24si@mahasiswa.pcr.ac.id arin_wbdl_2sib en Copyright © 1999-2024, Oracle and/or its affiliates. Oracle APEX 24.2.11</p> <p>Analisa: Query ini membuat tabel histAgents dengan struktur yang sama seperti tabel agents untuk menyimpan backup data agent yang dihapus, dilengkapi dengan kolom idhistA sebagai primary key dan tanggal untuk menyimpan timestamp penghapusan. Sequence idHisDelAg_seq dibuat untuk auto-generate ID, dan dapat diverifikasi dengan DESC histAgents untuk melihat struktur tabel atau SELECT dari user_tables untuk memastikan tabel berhasil dibuat.</p>	Object Type	TABLE	Object	HISTAGENTS	Table	IDHISTA	NUMBER	22	-	-	1	-	-	-		TANGGAL	TIMESTAMP(6)	11	-	6	-	✓	-	-		AGENT_CODE	CHAR	6	-	-	-	✓	-	-		AGENT_NAME	VARCHAR2	40	-	-	-	✓	-	-		WORKING_AREA	CHAR	35	-	-	-	✓	-	-		COMMISSION	NUMBER	-	12	2	-	✓	-	-		PHONE_NO	CHAR	15	-	-	-	✓	-	-		COUNTRY	VARCHAR2	25	-	-	-	✓	-	-
Object Type	TABLE	Object	HISTAGENTS																																																																																			
Table	IDHISTA	NUMBER	22	-	-	1	-	-	-																																																																													
	TANGGAL	TIMESTAMP(6)	11	-	6	-	✓	-	-																																																																													
	AGENT_CODE	CHAR	6	-	-	-	✓	-	-																																																																													
	AGENT_NAME	VARCHAR2	40	-	-	-	✓	-	-																																																																													
	WORKING_AREA	CHAR	35	-	-	-	✓	-	-																																																																													
	COMMISSION	NUMBER	-	12	2	-	✓	-	-																																																																													
	PHONE_NO	CHAR	15	-	-	-	✓	-	-																																																																													
	COUNTRY	VARCHAR2	25	-	-	-	✓	-	-																																																																													

2	<p>Buatlah sebuah trigger yang akan memindahkan data yang dihapus dari table agents ke table no-1! (Perhatikan relasi dari table agents)</p> <pre> CREATE OR REPLACE TRIGGER tDELAgent BEFORE DELETE ON agents FOR EACH ROW DECLARE v_count_orders NUMBER; v_count_customer NUMBER; BEGIN -- Backup data agents ke histAgents INSERT INTO histAgents VALUES (     idHisDelAg_seq.NEXTVAL,     LOCALTIMESTAMP,     :OLD.AGENT_CODE,     :OLD.AGENT_NAME,     :OLD.WORKING_AREA,     :OLD.COMMISSION,     :OLD.PHONE_NO,     :OLD.COUNTRY );  -- Hitung jumlah orders yang akan dihapus SELECT COUNT(*) INTO v_count_orders FROM orders WHERE AGENT_CODE = :OLD.AGENT_CODE;  -- Hitung jumlah customer yang akan dihapus SELECT COUNT(*) INTO v_count_customer FROM customer </pre>	<p>Trigger created.</p> <p>0.09 seconds</p>  <p>Analisa: Trigger tDELAgent berfungsi membackup data agent yang dihapus ke histAgents dan menangani relasi cascade dengan menghapus semua customer dan orders terkait agent tersebut. Trigger menghitung jumlah orders dan customer yang terkait, kemudian menghapusnya satu per satu (yang akan otomatis dibackup oleh trigger tDELOrder dan tDELCust), serta menampilkan informasi berapa banyak data yang dihapus dan dibackup melalui DBMS_OUTPUT, sehingga ketika agent 'A001' dihapus, semua data berelasi ikut terhapus dan tersimpan di tabel history masing-masing.</p>
---	---	---

```
WHERE AGENT_CODE =
:OLD.AGENT_CODE;

-- Hapus orders terkait agent
IF v_count_orders > 0 THEN
    DELETE FROM orders WHERE
AGENT_CODE = :OLD.AGENT_CODE;

DBMS_OUTPUT.PUT_LINE(v_count_orders
|| ' order(s) dari agent ' ||
:OLD.AGENT_CODE || ' dihapus dan
dibackup');
END IF;

-- Hapus customer terkait agent
IF v_count_customer > 0 THEN
    DELETE FROM customer WHERE
AGENT_CODE = :OLD.AGENT_CODE;

DBMS_OUTPUT.PUT_LINE(v_count_customer
|| ' customer(s) dari agent ' ||
:OLD.AGENT_CODE || ' dihapus dan
dibackup');
END IF;

DBMS_OUTPUT.PUT_LINE('Agent ' ||
:OLD.AGENT_CODE || '(' ||
:OLD.AGENT_NAME || ') berhasil dibackup
ke histAgents');
END;
/

DELETE FROM agents WHERE agent_code
= 'A001';
```

## KESIMPULAN

Praktikum ini memberikan pemahaman tentang konsep dan implementasi trigger dalam PL/SQL untuk automasi proses database. Trigger dapat digunakan untuk berbagai keperluan seperti menampilkan informasi data yang dimanipulasi (INSERT, UPDATE, DELETE), melakukan backup data otomatis sebelum penghapusan ke tabel history, dan menangani relasi antar tabel dengan cascade delete untuk menjaga integritas data. Melalui praktikum ini, dipelajari bahwa trigger memiliki jenis BEFORE dan AFTER yang eksekusinya tergantung timing yang dibutuhkan, serta dapat mengakses nilai data lama menggunakan :OLD dan data baru menggunakan :NEW. Penggunaan sequence sangat penting untuk generate primary key otomatis pada tabel backup/history. Trigger juga dapat dibuat sebagai compound trigger yang mendeteksi multiple operasi DML (INSERT, UPDATE, DELETE) dalam satu trigger menggunakan statement CASE. Dalam penerapannya, trigger sangat berguna untuk audit trail, data recovery, dan menjaga konsistensi data saat terjadi cascade operation pada tabel-tabel yang berelasi. Error yang sering terjadi adalah kesalahan penulisan nama kolom, tipe data yang tidak sesuai (VARCHAR vs VARCHAR2), dan ukuran field yang terlalu kecil, namun dapat diatasi dengan memahami struktur tabel dan tipe data Oracle SQL dengan benar. Secara keseluruhan, trigger merupakan fitur powerful dalam database yang membantu automasi proses bisnis dan menjaga kualitas data.