Final Database Project

BY- ISAAC TJERINA AND CHRIS KLEM
ELEMENTS OF DATABASES
DR. CANNATA
Explanation of Application Theme

- This application is a database that is intended for the use of employees at calculator company.
The Database’s Logical Model
Examples of Special Relations and Tables

- Many to Many Relation between Nov19_Product and Nov19_Orders with the association table, Nov19_Product_Orders, between them.

- Ternary Table Nov19_Shipping with one to many relations to Nov19_Product, Nov19_Order, and Nov19_Warehouse.
Example of Special Relations and Tables cont.

- Reference table Nov19_States
- Class and Subclass between Nov19_Components and its subclasses Nov19_Software and Nov19_Hardware
- One-to-one relationship between Nov19_Warehouse and Nov19_Manager
Data Model DDL Special Points

- Started all sequences at 100 to avoid any problems with the inserts

```sql
CREATE SEQUENCE Nov19_Clients_Client_ID_SEQ START WITH 100 NOCACHE ORDER;
CREATE OR REPLACE TRIGGER Nov19_Clients_Client_ID_TRG BEFORE INSERT ON Nov19_Clients FOR EACH ROW WHEN (NEW.Client_ID IS NULL) BEGIN :NEW.Client_ID := Nov19_Clients_Client_ID_SEQ.NEXTVAL;
END;
/

CREATE SEQUENCE Nov19_Components_Component_ID_SEQ START WITH 100 NOCACHE ORDER;
CREATE OR REPLACE TRIGGER Nov19_Components_Component_ID_TRG BEFORE INSERT ON Nov19_Components FOR EACH ROW WHEN (NEW.Component_ID IS NULL) BEGIN :NEW.Component_ID := Nov19_Components_Component_ID_SEQ.NEXTVAL;
END;
/
```
Views DDL Example

- Example showing the ddl for the Component view and trigger

```sql
create view Nov19_Components_view as
SELECT
  component_id,
  component_name,
  product_id,
  type,
  weight,
  material_id,
  language_id,
  gigabytes
FROM Nov19_Components where type = 'Nov19_Components';
```

```sql
create or replace TRIGGER Nov19_Components_trigger
INSTEAD OF insert ON Nov19_Components_view
FOR EACH ROW
BEGIN
  insert into Nov19_Components(
    component_id,
    component_name,
    product_id,
    type,
    weight,
    material_id,
    language_id,
    gigabytes)
  VALUES (;
END;
/`
```
Insets DML Example

- In this example the SQL is inserting into the Nov19_Hardware_View

```sql
INSERT INTO NOV19_HARDWARE_VIEW (COMPONENT_ID, COMPONENT_NAME, PRODUCT_ID, WEIGHT, MATERIAL_ID, TYPE) VALUES (6, 'Screen', 1, 0.31, 1, 'Nov19_Hardware');
INSERT INTO NOV19_HARDWARE_VIEW (COMPONENT_ID, COMPONENT_NAME, PRODUCT_ID, WEIGHT, MATERIAL_ID, TYPE) VALUES (7, 'CPU', 2, 5.24, 2, 'Nov19_Hardware');
INSERT INTO NOV19_HARDWARE_VIEW (COMPONENT_ID, COMPONENT_NAME, PRODUCT_ID, WEIGHT, MATERIAL_ID, TYPE) VALUES (8, 'Keyboard', 3, 7.6, 3, 'Nov19_Hardware');
INSERT INTO NOV19_HARDWARE_VIEW (COMPONENT_ID, COMPONENT_NAME, PRODUCT_ID, WEIGHT, MATERIAL_ID, TYPE) VALUES (9, 'Battery', 4, 2.21, 4, 'Nov19_Hardware');
INSERT INTO NOV19_HARDWARE_VIEW (COMPONENT_ID, COMPONENT_NAME, PRODUCT_ID, WEIGHT, MATERIAL_ID, TYPE) VALUES (10, 'Backlight', 5, 3.00, 5, 'Nov19_Hardware');
```
The “Number of Clients in Each State” region is a classic report that displays Number of Clients in each state and only shows the states with clients in them.

When the number in “Number of clients” column is selected the “Clients in States” region displays a classic report with the name and city that clients from the selected state are in (the picture show what is displayed when the 1 on DE row is selected).

The “Total Orders” Region displays a classic report that shows all of the clients and the total number of orders each client has made.
Apex Application: Clients Page cont.

- The “Number of Clients in Each State” classic report’s SQL

```
SELECT Name, STATE_ID, (SELECT Count(Client_ID) FROM Nov19_Clients c where c.State_ID = s.State_ID) as "Number of Clients"
FROM Nov19_States s
WHERE (SELECT Count(Client_ID) FROM Nov19_Clients c where c.State_ID = s.State_ID) > 0
```
Apex Application: Clients Page cont.

- The “Clients in State” classic report’s SQL:

```sql
SELECT c."CLIENT_ID", c."NAME", c."CITY"
FROM "OWNER"."MV18_CLIENTS" c
WHERE c.state_id = :PB_STATE
```

The result of the SQL query is displayed in the Clients in State table, showing the names and cities of clients in a specific state.
The “Total Orders” classic report’s SQL

```sql
SELECT Name, (SELECT COUNT(Order_ID) FROM Nov19_Orders o WHERE o.Client_ID = c.Client_ID) AS "Total Orders"
FROM Nov19_Clients c
```
The Managers Page is under the “Warehouse Info” tab.

The region on this page is an interactive report that displays the information for each manager.
The interactive report’s SQL

```
SELECT "MANAGER_ID",
"LAST_NAME",
"FIRST_NAME",
"WAREHOUSE_ID"
FROM ".\#OWNER\#\."NOV10_MANAGER"
```
The “Warehouses” Page is under the “Warehouse Info” tab. The chart region on this page is a chart, more specifically a pie chart, that displays the number of orders coming out of each warehouse. When a section of the pie chart is selected, the top right region displays an interactive report with the information about the manager of the warehouse selected. Also, when a section of the pie chart is selected, the “Orders” region displays a classic report with the information about the orders coming out of the selected region. The picture shows what is displayed when the 1 section of the pie chart is selected.
Apex Application: Warehouses

Page cont.

- The “Chart” chart’s SQL

```
1. SELECT null link, s.Warehouse_ID, COUNT(Order_ID) value
2. FROM Nov19_Shipping s JOIN Nov19_Warehouse w ON s.Warehouse_ID = w.Warehouse_ID
3. GROUP BY s.Warehouse_ID
```
Apex Application: Warehouses Page cont.

- The “Warehouse” interactive report’s SQL

```sql
SELECT w."WAREHOUSE_ID", w."Name" AS "Manager Name", w."City", w."State" AS "Location"
FROM "OWNER", "NOVI9_WAREHOUSE" w JOIN Novi9_States s ON w."State_ID" = s."State_ID"
JOIN Novi9_Manager m ON w."Manager_ID" = m."Manager_ID"
WHERE w."WAREHOUSE_ID" = :P12_WAREHOUSE_ID
```

<table>
<thead>
<tr>
<th>Manager name</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Estevez, Emilio</td>
<td>Gotham, NJ</td>
</tr>
</tbody>
</table>
The “Orders” classic report’s SQL

```sql
select o.ORDER_ID, o.COST, o.ORDER_DATE, c.NAME
from MOVIES_ORDERS o join nov19_shipping s on o.order_id = s.order_id
join mov19_clients c on o.client_id = c.client_id
where s.warehouse_id = :P12_WAREHOUSE_ID
```
Apex Application: Software Page

- The Software Page is under the “Components in Products” tab.
- The region on the top of the page is an interactive report that displays the information about each software component.
- The “Languages” region is a classic report that displays the languages that the software components are written in.
- The “Add a New Language” form region allows a user to add a language to the Nov19_Languages table.
The “Software” interactive report’s SQL
Apex Application: Software Page cont.

- The “Languages” classic report’s SQL

```
SELECT * 
FROM Nov19_Languages
```
Apex Application: Hardware Page

- The Hardware Page is under the “Components in Products” tab.
- The region on the top of the page is an interactive report that displays the information about each hardware component.
- The “Materials” region is a classic report that displays the materials that the hardware components are made in.
- The “Add a New Material” form region allows a user to add a language to the Nov19_Material table.
The “Hardware” interactive report’s SQL

```sql
SELECT "COMPONENT_ID", "Product_Name", "WEIGHT", Name
FROM "#OWNER#"."NOV19_HARDWARE_VIEW" h
JOIN Nov19_Material m ON h.Material_id = m.material_id
```

<table>
<thead>
<tr>
<th>Component Id</th>
<th>Weight</th>
<th>Product name</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>0.31</td>
<td>Sci-Calc 3000</td>
<td>Plastic</td>
</tr>
<tr>
<td>7</td>
<td>0.24</td>
<td>Calc 2.0</td>
<td>Steel</td>
</tr>
<tr>
<td>8</td>
<td>7.6</td>
<td>Calculator</td>
<td>Iron</td>
</tr>
<tr>
<td>9</td>
<td>2.21</td>
<td>Ledor</td>
<td>Copper</td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>Calcomatic</td>
<td>PVC</td>
</tr>
</tbody>
</table>
The "Materials" classic report's SQL:

```
SELECT *
FROM Nova9_Material
```
Apex Application: Shipping Page

- The region on the top of the page is an interactive report that displays the information from the ternary table.
- The “Orders not yet shipped” region displays a classic report that shows the orders that have not yet shipped.
Apex Application: Shipping Page cont.

- The “Shipping” interactive report's SQL

```sql
SELECT s.shipping_id, 
       w.city || ' ' || st.name AS "Warehouse Location", 
       c.name AS "Client Name", 
       p.product_name 
FROM Nova9_shipping s 
JOIN Nova9_warehouse w ON s.warehouse_id = w.warehouse_id 
JOIN Nova9_states st ON st.state_id = w.state_id 
JOIN Nova9_product p ON s.product_id = p.product_id 
LEFT JOIN Nova9_orders ord ON s.order_id = ord.order_id 
LEFT JOIN Nova9_clients c ON ord.client_id = c.client_id 
```
Apex Application: Shipping Page cont.

The “Orders not yet shipped” classic report’s SQL

```
Select Order_ID, Cost, Order_Date, Name
from Novia_Orders o join Novia_Clients c on o.Client_ID = c.Client_ID
where Order_ID not in (Select Order_ID from Novia_Shipping)
```
Apex Application: Shipping Form

Shared Components

► The Orders shared component’s SQL

```sql
select Order_id as d,
       Order_id as r
from Nov19_Orders
order by 1
```

► The Products shared component’s SQL

```sql
select product_name as d,
       Product_ID as r
from Nov19_Product
where Product_ID not in (Select Product_ID from Nov19_Shipping where Order_ID = :P31_Order_ID)
order by 1
```

► The Warehouse shared component’s SQL

```sql
select City as d,
       Warehouse_ID as r
from Nov19_Warehouse
where Warehouse_ID not in (Select Warehouse_ID from Nov19_Shipping where Order_ID = :P31_Order_ID and Product_ID = :P31_Product_ID)
order by 1
```
Apex Application: Orders Page

- The Orders Page is under the “Order info” tab
- The region on the top of the page is an interactive report that displays the information about each order
- The “Order Dates” region is a calendar that displays the orders on the days that they are ordered
Apex Application: Orders Page cont.

- The “Orders” interactive report’s SQL

```sql
select "ORDER_ID", "COST", Name, Order_date
from "#OWNER#"."NOV19_ORDERS" o join Nov19_Clients c on o.client_id = c.client_id
```
Apex Application: Orders Page cont.

- The “Order Date” calendar SQL

```sql
select NOV19_ORDERS.ORDER_DATE as ORDER_DATE, Order_ID
from NOV19_ORDERS
```
Apex Application: Products in Orders Page

- The Products in Orders Page is under the “Order info” tab.
- The region on the top of the page is an interactive report that displays the information about each product.
- The region on the bottom of the page is an interactive report that displays total amount of times each product has been ordered.

### Products in Orders

<table>
<thead>
<tr>
<th>Order Id</th>
<th>Product name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sci-Calc 3000</td>
</tr>
<tr>
<td>1</td>
<td>Calc 2.0</td>
</tr>
<tr>
<td>2</td>
<td>Calcomat</td>
</tr>
<tr>
<td>3</td>
<td>Leder</td>
</tr>
<tr>
<td>5</td>
<td>Leder</td>
</tr>
<tr>
<td>4</td>
<td>Calcomat</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Product Id</th>
<th>Product name</th>
<th>Price</th>
<th>Total amount ordered</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sci-Calc 3000</td>
<td>55.35</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Calc 2.0</td>
<td>24.35</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>Calcomat</td>
<td>60.86</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>Leder</td>
<td>24.42</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>Calcomat</td>
<td>53.35</td>
<td>2</td>
</tr>
</tbody>
</table>
Apex Application: Products in Orders Page cont.

The “Product Orders” interactive report’s SQL

```sql
SELECT Product_Name, "ORDER_ID"
FROM "owner"."NOV19_PRODUCT_ORDERS" 
  JOIN NOV19_Product p ON o.Product_id = p.Product_ID
```
Apex Application: Products in Orders Page cont.

- The “Product” interactive report’s SQL

```
SELECT p.Product_ID, Product_name, Price,
FROM Nov19_Product p
```