Midterm Project

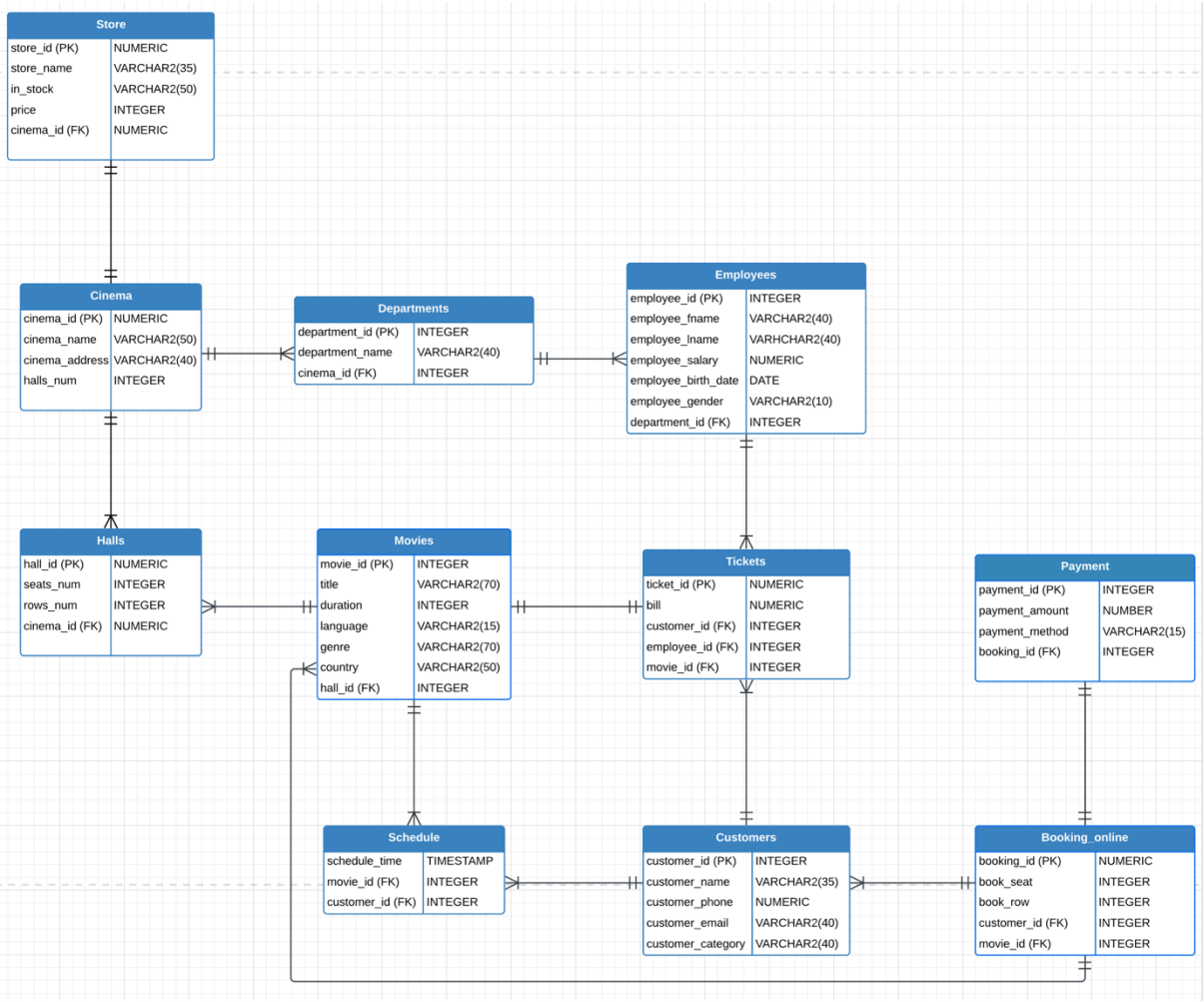
PL/SQL Programming 2

Cinema Database

Group: BDA-2006
Student: Merey Orazaly
Teacher: Raushania Tagauova

Nur-Sultan, 2022

Entity relationship diagram



Link

https://lucid.app/lucidchart/9f18f12a-7dcd-4832-9bf7-e5f378e6ab70/edit?invitationId=inv_15a03e6e-e68e-49d0-a67c-14d2ad08b8ea

Business rules:

1. Cinema has own store. (1:1)
2. Cinema has many departments. (1:M)
3. Department hire employees. (1:M)
4. Cinema has multiple halls. (1:M)
5. Movie can be shown in different halls, but hall can show only one movie. (1:M)
6. Customers selects one or many movies. (M:M)
7. Movies are shown according to schedule. (1:M)
8. Employees serve many customers. (M:M)
9. Each employee can sell many tickets. (1:M)
10. Customer can get many tickets. (1:M)
11. Only one ticket to one movie. (1:1)
12. Customer can book online. (1:M)
13. Only one payment is allowed to one booking. (1:1)
14. Customer can book many movies. (1:M)

Description of attributes with their entities

Attribute 1. Cinema

Field Name	Description	Type	Length
cinema_id (PK)	Cinema ID number	NUMERIC	
cinema_name	Cinema Name	VARCHAR2	50
cinema_address	Cinema Address	VARCHAR2	40
halls_num	Number of halls in cinema	INTEGER	

Attribute 2. Stores

Field Name	Description	Type	Length
store_id (PK)	Store ID number	NUMERIC	
store_name	Store Name	VARCHAR2	35
In_stock	Products available in store	VARCHAR2	50
price	Price of the products in store	INTEGER	
cinema_id (FK)	ID number of cinema store belongs to	NUMERIC	

Attribute 3. Halls

Field Name	Description	Type	Length
hall_id (PK)	Hall ID number	NUMERIC	
seats_num	Number of seats in hall	INTEGER	
rows_num	Number of rows in hall	INTEGER	
cinema_id (FK)	ID number of cinema hall belongs to	NUMERIC	

Attribute 4. Departments

Field Name	Description	Type	Length
department_id (PK)	Department ID number	INTEGER	
department_name	Department Name	VARCHAR2	40
cinema_id (FK)	ID number of cinema department belongs to	NUMERIC	

Attribute 5. Employees

Field Name	Description	Type	Length
employee_id (PK)	Employee ID number	INTEGER	
employee_fname	Employee First Name	VARCHAR2	40
employee_lname	Employee Last Name	VARCHAR2	40
employee_salary	Employee Salary	NUMERIC	
employee_birth_date	Employee Birth date	DATE	
employee_gender	Employee Gender	VARCHAR2	10
department_id (FK)	ID number of department, where employee works	INTEGER	

Attribute 6. Customers

Field Name	Description	Type	Length
customer_id (PK)	Customer ID number	INTEGER	
customer_name	Customer Name	VARCHAR2	35
customer_phone	Customer Phone Number	NUMERIC	
customer_email	Customer Email address	VARCHAR2	40
customer_category	E.g child/adult/student	VARCHAR2	40

Attribute 7. Movies

Field Name	Description	Type	Length
movie_id (PK)	Movie ID number	INTEGER	
title	Movie Title	VARCHAR2	70
duration	Movie Duration time	INTEGER	
language	Movie Language	VARCHAR2	15
genre	Movie Genre	VARCHAR2	70
country	Country shoots the movie	VARCHAR2	50
hall_id (FK)	ID number of hall showing the movie	INTEGER	

Attribute 8. Schedule – bridge (join) table

Field Name	Description	Type	Length
schedule_time	Time of showing movie	TIMESTAMP	
movie_id (FK)	ID number of movie show	INTEGER	
customer_id (FK)	ID number of watching customer	INTEGER	

Attribute 9. Tickets – bridge (join) table

Field Name	Description	Type	Length
ticket_id (PK)	Ticket ID number	NUMERIC	
bill	Bill of the show	NUMERIC	
customer_id (FK)	ID number of customer buying ticket	INTEGER	
employee_id (FK)	ID number of employee serving the customer's ticket	INTEGER	
movie_id (FK)	ID number of movie where customer need a ticket	INTEGER	

Attribute 10. Booking online

Field Name	Description	Type	Length
booking_id (PK)	Booking ID number	NUMERIC	
book_seat	Available seat number	INTEGER	
book_row	Available row number	INTEGER	
customer_id (FK)	ID number of customer booking movie	INTEGER	
movie_id (FK)	ID number of booking movie	INTEGER	

Attribute 11. Payment

Field Name	Description	Type	Length
payment_id (PK)	Payment ID number	NUMERIC	
payment_amount	Cash amount	INTEGER	
payment_method	E.g cash/card	VARCHAR2	30
Booking_id (FK)	ID number of booking to pay	NUMERIC	

Queries

```
----1. SQL query to get id, full name and salary of employees, who work at
department related with Human activity
SELECT employee_id, employee_fname || ' ' || emp.employee_lname AS
full_name, employee_salary
FROM employees emp
LEFT JOIN departments d on emp.department_id = d.department_id
WHERE d.department_name LIKE 'Human%';
```

Query Result

SQL | All Rows Fetched: 2 in 0,003 seconds

	EMPLOYEE_ID	FULL_NAME	EMPLOYEE_SALARY
1	18	Nick Fancourt	321530
2	42	Klaus Easson	371863

```
----2. SQL query to get number of male/female employees working in each
department
SELECT employee_gender, department_name, COUNT(*) AS employee_number
FROM employees
INNER JOIN departments d on d.department_id = employees.department_id
GROUP BY
GROUPING SETS (employee_gender), department_name
ORDER BY employee_gender DESC;
```

Query Result

SQL | All Rows Fetched: 18 in 0,004 seconds

	EMPLOYEE_GENDER	DEPARTMENT_NAME	EMPLOYEE_NUMBER
1	Male	Engineering	3
2	Male	Accounting	3
3	Male	Product Management	3
4	Male	Legal	2
5	Male	Human Resources	2
6	Male	Services	3
7	Male	Research and Development	2
8	Male	Sales	3
9	Male	Training	2
10	Male	Business Development	1
11	Female	Product Management	2
12	Female	Engineering	2
13	Female	Training	2
14	Female	Support	3
15	Female	Accounting	2
16	Female	Legal	2
17	Female	Business Development	4
18	Female	Research and Development	4

```
----3. SQL query to get employee full name, salary, gender from 15th
department and calculate the age using birth date column
SELECT employee_fname || ' ' || employee_lname AS full_name,
employee_salary, employee_gender, TRUNC(TO_NUMBER(SYSDATE -
TO_DATE(employee_birth_date)) / 365.25) AS AGE
FROM employees
NATURAL JOIN departments
WHERE department_id = 15;
```

Query Result

SQL | All Rows Fetched: 1 in 0,003 seconds

	FULL_NAME	EMPLOYEE_SALARY	EMPLOYEE_GENDER	AGE
1	Vale Grellier	367054	Female	23

```
----4. SQL query to get title, duration, language, genre, country, hall id
of movie shows
----scheduled between 20:00 and 23:00 in 14th April
SELECT title, duration, language, genre, country, hall_id
FROM movies
LEFT OUTER JOIN schedule s on movies.movie_id = s.movie_id
WHERE schedule_time BETWEEN '14-04-2022 20:00:00' AND '14-04-2022 23:00:00'
ORDER BY schedule_time;
```

Query Result

SQL | All Rows Fetched: 6 in 0,003 seconds

	TITLE	DURATION	LANGUAGE	GENRE	COUNTRY	HALL_ID
1	Safe in Hell	127	Italian	Drama	United States	443
2	West Point Story, The	140	Assamese	Comedy|Musical	South Africa	14
3	Murphy's Romance	107	Oriya	Comedy|Romance	Mexico	274
4	Last Remake of Beau Geste, The	115	Khmer	Adventure|Comedy	Philippines	373
5	Luxo Jr.	101	Haitian Creole	Animation|Children	United Kingdom	205
6	Violeta Went to Heaven (Violeta se fue a los cielos)	103	Māori	Drama	Vietnam	173

```
----5. SQL query to get name & address of cinema, title & bill of movie
show
----with bill price less than 1500 and duration less than 160 minutes
SELECT title, cinema_name, cinema_address, bill
FROM cinema
INNER JOIN halls h on cinema.cinema_id = h.cinema_id
INNER JOIN movies m on h.hall_id = m.hall_id
INNER JOIN tickets t on m.movie_id = t.movie_id
WHERE bill < 1500 AND duration < 160
ORDER BY bill DESC;
```

Query Result

SQL | All Rows Fetched: 8 in 0,005 seconds

	TITLE	CINEMA_NAME	CINEMA_ADDRESS	BILL
1	Violeta Went to Heaven (Violeta se fue a los cielos)	Hackett and Konopelski	9197 Shopko Hill	1427
2	Very Natural Thing, A	Chaplin	7212 Mega Silk Way	1322
3	Safe in Hell	Trantow, Leannon and Block	3426 Sutteridge Terrace	1308
4	Digging to China	Hermann-Rempel	0 Myrtle Lane	1251
5	American Grindhouse	Langosh, Medhurst and Glover	192 Bartillon Circle	1211
6	Beneath the Dark	Collier Group	0 Elmside Lane	1122
7	Dear God	Schaden Group	28733 Butterfield Place	998
8	Bad Company	Quitzon, Jacobson and Davis	512 Grover Place	847

```sql
----6. SQL query to get number of movies in unique genres
SELECT DISTINCT(genre), COUNT(movie_id)
FROM movies
GROUP BY genre;
```

Query Result

All Rows Fetched: 28 in 0,004 seconds

	GENRE	COUNT(MOVIE_ID)
1	Documentary	4
2	Crime|Drama|Film-Noir	2
3	Drama|Thriller	2
4	Adventure|Comedy	1
5	Animation|Children	1
6	Adventure|Animation|Comedy|Fantasy	1
7	Drama|Romance	1
8	Adventure|Documentary|Drama	1
9	Documentary|IMAX	1
10	Documentary|Mystery	1
11	Comedy|Crime	1
12	Action|Comedy	1
13	Action|Comedy|Crime	1
14	Comedy|Romance	2
15	Documentary|Musical	1
16	Crime|Drama|Horror|Thriller	1
17	Mystery|Thriller	1
18	Comedy|Drama|Romance	1
19	Action|Comedy|Crime|Thriller	1
20	Action|Crime|Drama	1
21	Drama|Mystery|Thriller	1
22	Comedy|Drama	1
23	Children|Comedy	2
24	Drama	8
25	Animation|Comedy|Drama	1
26	Comedy	4
27	Crime|Drama	1
28	Comedy|Musical	1

```sql
----7. SQL query to get information about 10 cheapest available seats for
movies except Murphy's Romance
----customer needs to know cinema name, title of movie, hall_id, available
seat, and price
SELECT cinema_name, title, hall_id, book_seat, book_row, payment_amount
FROM booking_online
NATURAL JOIN movies m
NATURAL JOIN payment p
NATURAL JOIN halls h
NATURAL JOIN cinema c
WHERE title NOT LIKE 'Murphy''s Romance'
ORDER BY payment_amount ASC
FETCH FIRST 10 ROWS ONLY;
```

Query Result

All Rows Fetched: 10 in 0,005 seconds

	CINEMA_NAME	TITLE	HALL_ID	BOOK_SEAT	BOOK_ROW	PAYMENT_AMOUNT
1	Thiel LLC	Invictus	81	11	8	832
2	Kshlerin-Hermiston	Soul of a Man, The	213	11	5	941
3	Quitzon, Jacobson and Davis	The Halloween That Almost Wasn't	304	6	13	942
4	Rodriguez-Daniel	Son, The (Le fils)	26	11	10	985
5	Collier Group	Beneath the Dark	202	6	12	1077
6	Konopelski-Feest	Women on the 6th Floor, The (Les Femmes du 6ème Étage)	354	1	12	1210
7	Little Group	Last Remake of Beau Geste, The	373	15	15	1338
8	Senger-Grant	West Point Story, The	14	13	8	1368
9	Langosh, Medhurst and Glover	Front Page, The	131	12	9	1461
10	Chaplin	Very Natural Thing, A	451	1	7	1518

```
----8. SQL query to get average bill price for different categories of
customers
SELECT customer_category, ROUND(AVG(bill),3)
FROM customers
RIGHT JOIN tickets t on customers.customer_id = t.customer_id
GROUP BY customer_category;
```

Query Result

SQL | All Rows Fetched: 3 in 0,003 seconds

	CUSTOMER_CATEGORY	ROUND(AVG(BILL),3)
1	Child	2183,846
2	Adult	1957,095
3	Student	1872,091

```
----9. SQL query to get information about maximum, minimum, average and
total salary of employees by departments
SELECT department_name, MAX(employee_salary) AS max_salary,
MIN(employee_salary) AS min_salary,
       ROUND(AVG(employee_salary), 2) AS average_salary,
SUM(employee_salary) AS total_salary
FROM employees
INNER JOIN departments d on d.department_id = employees.department_id
GROUP BY department_name;
```

Query Result

SQL | All Rows Fetched: 11 in 0,004 seconds

	DEPARTMENT_NAME	MAX_SALARY	MIN_SALARY	AVERAGE_SALARY	TOTAL_SALARY
1	Sales	196496	67786	122470	367410
2	Support	233504	100739	158909,67	476729
3	Engineering	390629	61456	231543,2	1157716
4	Product Management	402780	139093	265823	1329115
5	Research and Development	372455	92759	228432,5	1370595
6	Legal	308389	103980	250748,5	1002994
7	Training	367054	144955	225155	900620
8	Business Development	346154	52318	149118,2	745591
9	Services	365810	197504	258609	775827
10	Accounting	385940	178431	274734	1373670
11	Human Resources	371863	321530	346696,5	693393

```
----10. SQL query to get information about customers, who bought tickets
with bill higher than average bill price
----and we need to know who has served those customers
SELECT ticket_id, customer_name, customer_phone, customer_email, bill,
employee_fname
FROM tickets
INNER JOIN customers c on c.customer_id = tickets.customer_id
INNER JOIN employees e on e.employee_id = tickets.employee_id
WHERE bill > (SELECT AVG(bill) FROM tickets);
```

Query Result

SQL | All Rows Fetched: 24 in 0,005 seconds

	TICKET_ID	CUSTOMER_NAME	CUSTOMER_PHONE	CUSTOMER_EMAIL	BILL	EMPLOYEE_FNAME
1	3175456790	Lodovico	87771095082	lvero0@etsy.com	2956	Fee
2	597535930	Addy	87011359825	akrikorian3@meetup.com	2343	Melody
3	9185100323	Alexandre	87053208002	abilney4@topsy.com	2492	Brear
4	3895836184	Raynard	87475209804	rburdin7@vistaprint.com	2966	Horst
5	8844978334	Kippie	87052673435	kfern8@ibm.com	2416	Marty
6	4942867150	Ferd	87058965116	fteml9@livejournal.com	2488	Samson
7	513190724	Rochester	87029026761	rmonikerc@google.co.jp	2985	Lenna
8	2508674915	Grazia	87013492081	gnesfieldf@home.pl	2443	Skyler
9	5112522828	Netti	87474414795	nsigfridh@huffingtonpost.com	2167	Nick
10	4872218809	Zachery	87022612950	zparramorek@geocities.com	2445	Maryanne
11	1404038620	Janka	87087828470	jlipmann@i2i.jp	2172	Alon
12	9277081244	Idalia	87085254873	istaddono@t-online.de	2337	Renate
13	5969845752	Daron	87084536786	dtuftp@harvard.edu	2634	Cleavland
14	4190983829	Marketa	87088533899	mpietersenr@wunderground.com	2467	Burg
15	2098143451	Sheila-kathryn	87471393085	skidwellt@kickstarter.com	2385	Beniamino
16	2354474113	Leonanie	87785625009	lskamellu@blogs.com	2142	Prissie
17	4734121273	Consalve	87017116184	cguilbertz@ox.ac.uk	2279	Zondra
18	6941717114	Vallie	87011963468	vbeiderbecke10@youtu.be	2768	Stefano
19	7460139343	Bronny	87018317052	bwindybank11@smugmug.com	2847	Kelby
20	2304838553	Jerry	87012550358	jdoole12@nyu.edu	2008	Juliane
21	2107210802	Niko	87015544690	nhurworth13@comcast.net	2779	Valentijn
22	8280720219	Carlee	87789653359	ctrodden15@army.mil	2258	Klaus
23	5682182154	Cirstoforo	87781605768	cmcure16@edublogs.org	2271	Issy
24	4904971450	Garik	87754271950	ghallan18@wordpress.com	2385	Odie

Procedures and Functions

```
----1. Procedure that will raise employee's salary. Ask user to enter
employee_id.
--If employee is female from Accounting department, then raise salary by
10%. Otherwise rollback.
SELECT employee_id, employee_salary FROM employees
INNER JOIN departments d on employees.department_id = d.department_id
WHERE employee_gender = 'Female' AND department_name = 'Accounting';

CREATE OR REPLACE Procedure raise_salaries(emp_id int)
IS
    dep VARCHAR2(70);
    female VARCHAR2(70);
BEGIN
    SELECT employee_gender INTO female FROM employees
    WHERE employee_id = emp_id;

    SELECT department_name INTO dep FROM departments
    INNER JOIN employees e on departments.department_id = e.department_id
    WHERE employee_id = emp_id;

    IF (female = 'Female' AND dep = 'Accounting') THEN
        UPDATE employees
        SET employee_salary = employee_salary + (employee_salary * 0.1)
        WHERE employee_id = emp_id;

        DBMS_OUTPUT.PUT_LINE('Employee''s salary with id ' || emp_id || '
raised');
        COMMIT;
    ELSE
        DBMS_OUTPUT.PUT_LINE('Employee is not female from accounting
department');
        ROLLBACK;
    END IF;
END;
/

CALL raise_salaries(25);
```

Script Output

Task completed in 0,03 seconds

```
EMPLOYEE_ID EMPLOYEE_SALARY
----------- ---------------
         11          351712
         19          353410


Procedure RAISE_SALARIES compiled

Employee is not female from accounting department


Call completed.
```

```
---2. Function to calculate total salary of all employees
CREATE OR REPLACE FUNCTION total_salary
RETURN number IS
   total number;
BEGIN
   SELECT SUM(employee_salary) into total
   FROM employees;

    RETURN total;
END total_salary;
/

DECLARE
   total number;
BEGIN
   total := total_salary();
   DBMS_OUTPUT.PUT_LINE('Total salary: ' || total);
END;
/
```

Script Output

Task completed in 0,037 seconds

```
Function TOTAL_SALARY compiled

Total salary: 9904023


PL/SQL procedure successfully completed.
```

```
----3. Procedure to find employee under 21 and delete him from cinema db.
SELECT employee_fname || ' ' || employee_lname AS full_name,
employee_salary, employee_gender, TRUNC(TO_NUMBER(SYSDATE -
TO_DATE(employee_birth_date)) / 365.25) AS AGE
FROM employees
WHERE TRUNC(TO_NUMBER(SYSDATE - TO_DATE(employee_birth_date)) / 365.25) <
21;

CREATE OR REPLACE Procedure delete_juniors
IS
    adult int;
BEGIN
    SELECT employee_id
    INTO adult
    FROM employees
    WHERE TRUNC(TO_NUMBER(SYSDATE - TO_DATE(employee_birth_date)) / 365.25)
< 21
    FETCH FIRST 1 ROWS ONLY;

    DELETE FROM tickets
    WHERE employee_id IN (SELECT employee_id FROM employees WHERE
employee_id = adult);

    DELETE FROM employees
    WHERE employee_id IN (SELECT employee_id FROM employees WHERE
employee_id = adult);

    DBMS_OUTPUT.PUT_LINE('Employee under 21 was deleted from db');
END;
/
```

```
CALL delete_juniors();
```

Script Output

Task completed in 0,038 seconds

```
FULL_NAME                                                                          EMPLOYEE_SALARY EMPLOYEE_G        AGE
---------------------------------------------------------------------------------- --------------- ---------- ----------
Kelby Whitters                                                                              295562 Male               20
Valentijn Latliff                                                                            67786 Male               20
Klaus Easson                                                                                371863 Male               20


Procedure DELETE_JUNIORS compiled

Employee under 21 was deleted from db


Call completed.

FULL_NAME                                                                          EMPLOYEE_SALARY EMPLOYEE_G        AGE
---------------------------------------------------------------------------------- --------------- ---------- ----------
Valentijn Latliff                                                                            67786 Male               20
Klaus Easson                                                                                371863 Male               20
```

```
----4. Function to calculate age of particular employee
CREATE OR REPLACE FUNCTION calculate_age(employeeId NUMBER)
RETURN NUMBER IS
    age NUMBER;
BEGIN
    SELECT TRUNC(TO_NUMBER(SYSDATE - TO_DATE(employee_birth_date)) /
365.25) INTO age
    FROM employees WHERE employee_id = employeeId;
    RETURN age;
END calculate_age;
/

SELECT employee_fname || ' ' || employee_lname AS full_name,
employee_gender, calculate_age(15) AS age
FROM employees
WHERE employee_id = 15;
```

Script Output

Task completed in 0,057 seconds

```
Function CALCULATE_AGE compiled


FULL_NAME                                                                          EMPLOYEE_G        AGE
---------------------------------------------------------------------------------- ---------- ----------
Vale Grellier                                                                      Female             23
```

```
----5. Procedure that insert new department if it doesn't exist.
CREATE OR REPLACE PROCEDURE insert_dep
   (dep_name IN VARCHAR2)
IS
   dep_id number;
   max_dept number;

   CURSOR c1 IS
   SELECT department_id FROM departments
   WHERE department_name = dep_name;
BEGIN
   OPEN c1;
   FETCH c1 INTO dep_id;

   SELECT MAX(department_id) + 1 INTO max_dept FROM departments;

   IF c1%NOTFOUND THEN
```

```
      dep_id := max_dept;
   END IF;

    INSERT INTO departments (department_id, department_name, cinema_id)
    VALUES (dep_id, dep_name, 1);

   COMMIT;

   CLOSE c1;
END;

CALL insert_dep('Cashiers');

SELECT*FROM departments WHERE department_name = 'Cashiers';
```

Script Output

Task completed in 0,038 seconds

```
           36 Business Development                                 36
           37 Sales                                                37
           38 Product Management                                   38
           39 Research and Development                             39
           40 Sales                                                40
           41 Support                                              41
           42 Human Resources                                      42
           43 Research and Development                             43
           44 Training                                             44

DEPARTMENT_ID DEPARTMENT_NAME                           CINEMA_ID
------------- ---------------------------------------- ----------
           45 Business Development                              45

45 rows selected.


Procedure INSERT_DEP compiled


Call completed.


DEPARTMENT_ID DEPARTMENT_NAME                           CINEMA_ID
------------- ---------------------------------------- ----------
           46 Cashiers                                           1
```