Lab report #4

Sadovskaya Veronika

GitHub: https://github.com/sdveronika/DataMola22

Create all necessary tables for the cleansing area from the DW_CL user in the ts_dw_cl tablespace:

1) create table dw_cl.cl_t_client

```
21 CREATE TABLE dw_cl.cl_t_client(
22  first_name VARCHAR2(20) NOT NULL,
23  last_name VARCHAR2(20) NOT NULL,
24  phone VARCHAR2(20) NOT NULL,
25  email VARCHAR2(40) NOT NULL,
26  street VARCHAR2(40) NOT NULL,
27  country VARCHAR2(20) NOT NULL,
28  city VARCHAR2(20) NOT NULL,
29  client_status CHAR(1)NOT NULL CHECK (client_status IN ('N','Y'))
30  )TABLESPACE ts_dw_cl;
```

Script Output | Task completed in 0,119 seconds

```
Table DW_CL.CL_T_CLIENT created.
```

2) create table dw_cl.cl_t_dish

```
36 CREATE TABLE dw_cl.cl_t_dish(
37  dish_name VARCHAR2(20) NOT NULL,
38  dish_category VARCHAR2(20) NOT NULL,
39  price DECIMAL (10,2) NOT NULL,
40  composition VARCHAR2(150) NOT NULL,
41  weight DECIMAL (10,2) NOT NULL,
42  dish_status CHAR(1) NOT NULL CHECK (dish_status IN ('N','Y'))
43  )TABLESPACE ts_dw_cl;
44
```

Script Output | Task completed in 0,057 seconds

```
Table DW_CL.CL_T_DISH created.
```

3) create table dw_cl.cl_t_employee

```
49 CREATE TABLE dw_cl.cl_t_employee(
50  first_name VARCHAR2(20) NOT NULL,
51  last_name VARCHAR2(20) NOT NULL,
52  phone VARCHAR2(20) NOT NULL,
53  email VARCHAR2(40) NOT NULL,
54  department VARCHAR2(20) NOT NULL,
55  job_title VARCHAR2(20)NOT NULL,
56  street VARCHAR2(40) NOT NULL,
57  country VARCHAR2(20) NOT NULL,
58  city VARCHAR2(20) NOT NULL,
59  building INT NOT NULL,
60  apartment INT NOT NULL,
61  employee_status CHAR(1) NOT NULL CHECK (employee_status IN ('N','Y')))
62  TABLESPACE ts_dw_cl;
```

Script Output | Task completed in 0,07 seconds

```
Table DW_CL.CL_T_EMPLOYEE created.
```

4) create table dw_cl.cl_t_gen_period

```
68 CREATE TABLE dw_cl.cl_t_gen_period (
69      valid_from      DATE NOT NULL,
70      valid_to        DATE NOT NULL,
71      promotion_name  VARCHAR2(100) NOT NULL,
72      promotion_percent DECIMAL (4,2) NOT NULL,
73      decription      VARCHAR2(100) NOT NULL
74  )TABLESPACE ts_dw_cl;
```

Script Output | Task completed in 0,053 seconds

```
Table DW_CL.CL_T_GEN_PERIOD created.
```

5) create table dw_cl.cl_t_payment_method

```
80  CREATE TABLE dw_cl.cl_t_payment_method(
81  payment_method_name VARCHAR2(20) NOT NULL,
82  payment_method_status CHAR(1) NOT NULL CHECK (payment_method_status IN ('N','Y'))
83  ) TABLESPACE ts_dw_cl;
```

Script Output | Task completed in 0,052 seconds

```
Table DW_CL.CL_T_PAYMENT_METHOD created.
```

6) create table dw_cl.cl_t_restaurant

```
89  CREATE TABLE dw_cl.cl_t_restaurant(
90  phone VARCHAR2(20) NOT NULL,
91  email VARCHAR2(40) NOT NULL,
92  street VARCHAR2(40) NOT NULL,
93  country VARCHAR2(20) NOT NULL,
94  city VARCHAR2(20) NOT NULL,
95  building INT NOT NULL,
96  apartment INT NOT NULL,
97  restaurant_status CHAR(1) NOT NULL CHECK (restaurant_status IN ('N','Y'))
98  )TABLESPACE ts_dw_cl;
```

Script Output | Task completed in 0,066 seconds

```
Table DW_CL.CL_T_RESTAURANT created.
```

7) create table dw_cl.cl_t_transaction

```
106 CREATE TABLE dw_cl.cl_t_transaction(
107 --
108 first_name_c VARCHAR2(20) NOT NULL,
109 last_name_c VARCHAR2(20) NOT NULL,
110 phone_c VARCHAR2(20) NOT NULL,
111 email_c VARCHAR2(40) NOT NULL,
112 street_c VARCHAR2(40) NOT NULL,
113 country_c VARCHAR2(20) NOT NULL,
114 city_c VARCHAR2(20) NOT NULL,
115 client_status CHAR(1) NOT NULL CHECK (client_status IN ('N','Y')),
116 --
117 dish_name VARCHAR2(20) NOT NULL,
118 dish_category VARCHAR2(20) NOT NULL,
119 price DECIMAL (10,2) NOT NULL,
120 composition VARCHAR2(150) NOT NULL,
121 weight DECIMAL (10,2) NOT NULL,
122 dish_status CHAR(2) NOT NULL CHECK (dish_status IN ('N','Y')),
123 --
124 phone_r VARCHAR2(20) NOT NULL,
```

Script Output | Task completed in 0,161 seconds

```
Table DW_CL.CL_T_TRANSACTION created.
```

Create all necessary stored procedure for moving all data from storage area to cleansing layer. The main task of these procedures is to get rid of invalid data (that is, remove null values).

1) create package pkg_etl_clients_cl and stored procedure load_clean_clients

```
1  CREATE OR REPLACE PACKAGE pkg_etl_clients_cl
2  AS
3      PROCEDURE load_CLEAN_CLIENTS;
4  END pkg_etl_clients_cl;
```

Script Output | Task completed in 0,133 seconds

```
Package PKG_ETL_CLIENTS_CL compiled
```

```
6  CREATE OR REPLACE PACKAGE body pkg_etl_clients_cl
7  AS
8    PROCEDURE load_CLEAN_CLIENTS
9     AS
10       CURSOR c_v IS
11          SELECT DISTINCT first_name
12                        , last_name
13                        , phone
14                        , email
15                        , street
16                        , country
17                        , city
18                        , client_status
19            FROM sa_clients.sa_t_client
20            WHERE first_name IS NOT NULL
21            AND last_name IS NOT NULL
22            AND phone IS NOT NULL
23            AND email IS NOT NULL
```

Script Output | Query Result | Task completed in 0,612 seconds

```
Package Body PKG_ETL_CLIENTS_CL compiled
```

2) create package pkg_etl_dishes_cl and stored procedure load_clean_dishes

```
1  CREATE OR REPLACE PACKAGE pkg_etl_dishes_cl
2  AS
3     PROCEDURE load_CLEAN_DISHES;
4  END pkg_etl_dishes_cl;
```

Script Output | Task completed in 0,101 seconds

```
Package PKG_ETL_DISHES_CL compiled
```

```
5
6  CREATE OR REPLACE PACKAGE body pkg_etl_dishes_cl
7  AS
8    PROCEDURE load_CLEAN_DISHES
9     AS
10       CURSOR c_v1 IS
11          SELECT  dish_name
12                        , dish_category
13                        , price
14                        , composition
15                        , weight
16                        , dish_status
17            FROM  sa_restaurants.sa_t_dish
18            WHERE dish_name IS NOT NULL
19            AND dish_category IS NOT NULL
20            AND price IS NOT NULL
21            AND composition IS NOT NULL
22            AND weight IS NOT NULL
23            AND dish_status IS NOT NULL;
24    BEGIN
```

Script Output | Query Result | Task completed in 0,587 seconds

```
Package Body PKG_ETL_DISHES_CL compiled
```

3) create package pkg_etl_employees_cl and stored procedure load_clean_employees

```
1  CREATE OR REPLACE PACKAGE pkg_etl_employees_cl
2  AS
3      PROCEDURE load_CLEAN_EMPLOYEES;
4  END pkg_etl_employees_cl;
```

Script Output | Task completed in 0,121 seconds

```
Package PKG_ETL_EMPLOYEES_CL compiled
```

```
6  CREATE OR REPLACE PACKAGE body pkg_etl_employees_cl
7  AS
8    PROCEDURE load_CLEAN_EMPLOYEES
9     AS
10       CURSOR c_v IS
11          SELECT DISTINCT first_name
12                        , last_name
13                        , phone
14                        , email
15                        , department
16                        , job_title
17                        , street
18                        , country
19                        , city
20                        , building
21                        , apartment
22                        , employee_status
23              FROM sa_restaurants.sa_t_employee
24              WHERE first_name IS NOT NULL
```

Script Output | Query Result | Task completed in 0,602 seconds

```
Package Body PKG_ETL_EMPLOYEES_CL compiled
```

4) create package pkg_etl_gen_priods_cl and stored procedure load_clean_gen_periods

```
1  CREATE OR REPLACE PACKAGE pkg_etl_gen_priods_cl
2  AS
3      PROCEDURE load_CLEAN_GEN_PERIODS;
4  END pkg_etl_gen_priods_cl;
```

Script Output | Task completed in 0,103 seconds

```
Package PKG_ETL_GEN_PRIODS_CL compiled
```

```
8    PROCEDURE load_CLEAN_GEN_PERIODS
9     AS
10       CURSOR c_v IS
11          SELECT DISTINCT valid_from
12                        , valid_to
13                        , promotion_name
14                        , promotion_percent
15                        , decription
16            FROM sa_restaurants.sa_t_gen_period
17            WHERE valid_from IS NOT NULL
18            AND valid_to IS NOT NULL
19            AND promotion_name IS NOT NULL
20            AND promotion_percent IS NOT NULL
21            AND decription IS NOT NULL;
22     BEGIN
23     EXECUTE IMMEDIATE 'TRUNCATE TABLE dw_cl.cl_t_gen_period';
24        FOR i IN c_v LOOP
25           INSERT INTO dw_cl.cl_t_gen_period(
26                       valid_from
```

Script Output | Query Result | Task completed in 0,597 seconds

```
Package Body PKG_ETL_GEN_PRIODS_CL compiled
```

5) create package pkg_etl_payment_methods_cl and stored procedure load_clean_payment_methods

```
CREATE OR REPLACE PACKAGE pkg_etl_payment_methods_cl
AS
   PROCEDURE load_CLEAN_PAYMENT_METHODS;
END pkg_etl_payment_methods_cl;
```

Script Output | Task completed in 0,115 seconds

```
Package PKG_ETL_PAYMENT_METHODS_CL compiled
```

```
CREATE OR REPLACE PACKAGE body pkg_etl_payment_methods_cl
AS
  PROCEDURE load_CLEAN_PAYMENT_METHODS
   AS
      CURSOR c_v IS
         SELECT DISTINCT payment_method_name
                       , payment_method_status
           FROM sa_restaurants.sa_t_payment_method
           WHERE payment_method_name IS NOT NULL
           AND payment_method_status IS NOT NULL;
   BEGIN
   EXECUTE IMMEDIATE 'TRUNCATE TABLE dw_cl.cl_t_payment_method';
      FOR i IN c_v LOOP
         INSERT INTO dw_cl.cl_t_payment_method(
                        payment_method_name
                       , payment_method_status)
              VALUES ( i.payment_method_name
                       , i.payment_method_status );
         EXIT WHEN c_v%NOTFOUND;
```

Script Output | Query Result | Task completed in 0,566 seconds

```
Package Body PKG_ETL_PAYMENT_METHODS_CL compiled
```

6) create package pkg_etl_restaurants_cl and stored procedure load_clean_restaurants

```
CREATE OR REPLACE PACKAGE pkg_etl_restaurants_cl
AS
   PROCEDURE load_CLEAN_RESTAURANTS;
END pkg_etl_restaurants_cl;
```

Script Output | Task completed in 0,095 seconds

```
Package PKG_ETL_RESTAURANTS_CL compiled
```

```
CREATE OR REPLACE PACKAGE body pkg_etl_restaurants_cl
AS
  PROCEDURE load_CLEAN_RESTAURANTS
   AS
      CURSOR c_v IS
         SELECT DISTINCT phone
                       , email
                       , street
                       , country
                       , city
                       , building
                       , apartment
                       , restaurant_status
           FROM sa_restaurants.sa_t_restaurant
           WHERE phone IS NOT NULL
           AND email IS NOT NULL
           AND street IS NOT NULL
           AND country IS NOT NULL
           AND street IS NOT NULL
```

Script Output | Query Result | Task completed in 0,594 seconds

```
Package Body PKG_ETL_RESTAURANTS_CL compiled
```

7) create package pkg_etl_transactions_cl and stored procedure load_clean_transactions

```
1  CREATE OR REPLACE PACKAGE pkg_etl_transactions_cl
2  AS
3     PROCEDURE load_CLEAN_TRANSACTIONS;
4  END pkg_etl_transactions_cl;
```

Script Output | Task completed in 0,128 seconds

```
Package PKG_ETL_TRANSACTIONS_CL compiled
```

```
6  CREATE OR REPLACE PACKAGE body pkg_etl_transactions_cl
7  AS
8    PROCEDURE load_CLEAN_TRANSACTIONS
9     AS
10       CURSOR c_v IS
11          SELECT DISTINCT first_name_c
12                        , last_name_c
13                        , phone_c
14                        , email_c
15                        , street_c
16                        , country_c
17                        , city_c
18                        , client_status
19                        , dish_name
20                        , dish_category
21                        , price
22                        , composition
23                        , weight
```

Script Output | Query Result | Task completed in 1,147 seconds

```
Package Body PKG_ETL_TRANSACTIONS_CL compiled
```

Run all procedures:

```
1  BEGIN
2      pkg_etl_clients_cl.load_CLEAN_CLIENTS;
3      pkg_etl_dishes_cl.load_CLEAN_DISHES;
4      pkg_etl_employees_cl.load_CLEAN_EMPLOYEES;
5      pkg_etl_gen_priods_cl.load_CLEAN_GEN_PERIODS;
6      pkg_etl_payment_methods_cl.load_CLEAN_PAYMENT_METHODS;
7      pkg_etl_restaurants_cl.load_CLEAN_RESTAURANTS;
8      pkg_etl_transactions_cl.load_CLEAN_TRANSACTIONS;
9  END;
```

Script Output | Task completed in 28,355 seconds

```
PL/SQL procedure successfully completed.
```

Select data from tables on cleansing level:

1) dw_cl.cl_t_client

```
12  SELECT * FROM dw_cl.cl_t_client;
```

Query Result | SQL | All Rows Fetched: 5 in 0,042 seconds

	FIRST_NAME	LAST_NAME	PHONE	EMAIL	STREET	COUNTRY	CITY	CLIENT_STATUS
1	Adriana	Karnitskaya	375294217925	AdrianaKarnitskaya@mail.ru	Yesenin	Belarus	Minsk	Y
2	Ada	Alymova	375292376362	AdaAlymova@mail.ru	Nikolskaya	Russia	Moscow	Y
3	Alice	Lysenko	375294487084	AliceLysenko@mail.ru	Fifth Avenue	USA	New York	Y
4	Ian	Astafyev	375295321165	IanAstafyev@mail.ru	Champs Elysees	France	Paris	Y
5	Pavel	Moshko	375291330060	PavelMoshko@mail.ru	Antoine Dansaert	Belgium	Brussels	Y

2) dw_cl. cl_t_dish

```
13  SELECT * FROM dw_cl.cl_t_dish;
```

Query Result | SQL | All Rows Fetched: 5 in 0,043 seconds

	DISH_NAME	DISH_CATEGORY	PRICE	COMPOSITION	WEIGHT	DISH_STATUS
1	chebupelli	hot	61	chebupelli ingredients	512	Y
2	pasta	hot	87	pasta ingredients	959	Y
3	soup	hot	15	soup ingredients	915	Y
4	pizza	hot	25	pizza ingredients	486	Y
5	greek salad	appetizer	82	greek salad ingredients	797	Y

3) dw_cl. cl_t_employee

```
14  SELECT * FROM dw_cl.cl_t_employee;
```

Query Result | SQL | All Rows Fetched: 25 in 0,048 seconds

	FIRST_NAME	LAST_NAME	PHONE	EMAIL	DEPARTMENT	JOB_TITLE	STREET	COUNTRY	CITY	BUILDI
1	Adriana	Karnitskaya	375291698908	AdrianaKarnitskaya@mail.ru	department_name_1	director	Yesenin	Belarus	Minsk	
2	Ada	Alymova	375299826609	AdaAlymova@mail.ru	department_name_2	manager	Nikolskaya	Russia	Moscow	
3	Alice	Lysenko	375298183473	AliceLysenko@mail.ru	department_name_3	employee	Fifth Avenue	USA	New York	
4	Alyssa	Malysheva	375290801452	AlyssaMalysheva@mail.ru	department_name_4	employee	Vladimirskaya	Ukraine	Kiev	
5	Anna	Buynova	375290565078	AnnaBuynova@mail.ru	department_name_5	employee	Gurchevskaya	Poland	Warsaw	
6	Christina	Mayorova	375292764881	ChristinaMayorova@mail.ru	department_name_1	director	Gammel Strand	Denmark	Copenhagen	
7	Clara	Zaykova	375295778867	ClaraZaykova@mail.ru	department_name_2	manager	Abby	England	London	
8	Veronika	Sadovskaya	375296747159	VeronikaSadovskaya@mail.ru	department_name_3	employee	D Dunkri	Estonia	Tallinn	
9	Eva	Moshko	375292569304	EvaMoshko@mail.ru	department_name_4	employee	Aloyas	Latvia	Riga	
10	Vera	Lebedeva	375299493212	VeraLebedeva@mail.ru	department_name_5	employee	G Galve	Lithuania	Vilnius	
11	Alexandra	Etkina	375295186852	AlexandraEtkina@mail.ru	department_name_1	director	Graben	Austria	Vienna	
12	Rita	Astafyeva	375299300168	RitaAstafyeva@mail.ru	department_name_2	manager	K Kaiser-Friedrich	Germany	Berlin	
13	Ian	Astafyev	375297942988	IanAstafyev@mail.ru	department_name_3	employee	Champs Elysees	France	Paris	
14	Pavel	Moshko	375296156608	PavelMoshko@mail.ru	department_name_4	employee	Antoine Dansaert	Belgium	Brussels	

4) dw_cl. cl_t_gen_period

```
15  SELECT * FROM dw_cl.cl_t_gen_period;
```

Query Result | SQL | All Rows Fetched: 6 in 0,037 seconds

	VALID_FROM	VALID_TO	PROMOTION_NAME	PROMOTION_PERCENT	DECRIPTION
1	01.01.21	31.01.21	promotion_name_1	25	decription_1
2	01.02.21	28.02.21	promotion_name_2	15	decription_2
3	01.03.21	31.03.21	promotion_name_3	10	decription_3
4	01.04.21	30.04.21	promotion_name_4	5	decription_4
5	01.05.21	31.05.21	promotion_name_5	20	decription_5
6	01.06.21	31.12.21	promotion_name_6	20	decription_6

5) dw_cl. cl_t_payment_method

```
16  SELECT * FROM dw_cl.cl_t_payment_method;
```

Query Result | SQL | All Rows Fetched: 2 in 0,037 seconds

	PAYMENT_METHOD_NAME	PAYMENT_METHOD_STATUS
1	bank card	Y
2	cash	Y

6) dw_cl. cl_t_restaurant

```
17  SELECT * FROM dw_cl.cl_t_restaurant;
```

Query Result | SQL | All Rows Fetched: 5 in 0,038 seconds

	PHONE	EMAIL	STREET	COUNTRY	CITY	BUILDING	APARTMENT	RESTAURANT_STATUS
1	375294308850	1@mail.ru	Yesenin	Belarus	Minsk	81	132	Y
2	375299013465	2@mail.ru	Nikolskaya	Russia	Moscow	32	165	Y
3	375292356572	3@mail.ru	Fifth Avenue	USA	New York	30	189	Y
4	375295176493	4@mail.ru	Vladimirskaya	Ukraine	Kiev	36	185	Y
5	375295841669	5@mail.ru	Gurchevskaya	Poland	Warsaw	46	230	Y

7) dw_cl. cl_t_transaction

```
SELECT * FROM dw_cl.cl_t_transaction;
```

Query Result — Fetched 50 rows in 0,065 seconds

	FIRST_NAME_C	LAST_NAME_C	PHONE_C	EMAIL_C	STREET_C	COUNTRY_C	CITY_C	CLIENT_STATUS	DISH_NAME	DISH_
1	Alice	Lysenko	375299430223	AliceLysenko@mail.ru	Fifth Avenue	USA	New York	Y	chebupelli	hot
2	Ada	Alymova	375294588894	AdaAlymova@mail.ru	Nikolskaya	Russia	Moscow	Y	pizza	hot
3	Alice	Lysenko	375299430223	AliceLysenko@mail.ru	Fifth Avenue	USA	New York	Y	pizza	hot
4	Ian	Astafyev	375297617914	IanAstafyev@mail.ru	Champs Elysees	France	Paris	Y	pizza	hot
5	Ada	Alymova	375294588894	AdaAlymova@mail.ru	Nikolskaya	Russia	Moscow	Y	chebupelli	hot
6	Ian	Astafyev	375297617914	IanAstafyev@mail.ru	Champs Elysees	France	Paris	Y	soup	hot
7	Pavel	Moshko	375292729548	PavelMoshko@mail.ru	Antoine Dansaert	Belgium	Brussels	Y	pizza	hot
8	Alice	Lysenko	375299430223	AliceLysenko@mail.ru	Fifth Avenue	USA	New York	Y	soup	hot
9	Ian	Astafyev	375297617914	IanAstafyev@mail.ru	Champs Elysees	France	Paris	Y	chebupelli	hot
10	Alice	Lysenko	375299430223	AliceLysenko@mail.ru	Fifth Avenue	USA	New York	Y	pasta	hot
11	Alice	Lysenko	375299430223	AliceLysenko@mail.ru	Fifth Avenue	USA	New York	Y	pizza	hot
12	Adriana	Karnitskaya	375290621595	AdrianaKarnitskaya@mail.ru	Yesenin	Belarus	Minsk	Y	chebupelli	hot
13	Ada	Alymova	375294588894	AdaAlymova@mail.ru	Nikolskaya	Russia	Moscow	Y	chebupelli	hot
14	Alice	Lysenko	375299430223	AliceLysenko@mail.ru	Fifth Avenue	USA	New York	Y	soup	hot

Create all necessary tables and sequences for the DW layer from the DW_DATA user in the ts_dw_data_01 tablespace:

1) create table dw_data.client_dimension and sequence seq_client_dim

```
159 CREATE TABLE dw_data.client_dimension(
160 client_id NUMBER,
161 first_name VARCHAR2(20) NOT NULL,
162 last_name VARCHAR2(20) NOT NULL,
163 phone VARCHAR2(20) NOT NULL,
164 email VARCHAR2(40) NOT NULL,
165 street VARCHAR2(40) NOT NULL,
166 country VARCHAR2(20) NOT NULL,
167 city VARCHAR2(20) NOT NULL,
168 status CHAR(1) NOT NULL CHECK (status IN ('N','Y')),
169 CONSTRAINT CLIENT_UNIQUE UNIQUE (phone, email),
170 CONSTRAINT CLIENT_ID_PK PRIMARY KEY ( client_id ) ENABLE);
```

Script Output — Task completed in 0,079 seconds

```
Table DW_DATA.CLIENT_DIMENSION created.
```

```
173 CREATE SEQUENCE SEQ_CLIENT_DIM
174  START WITH      1
175  INCREMENT BY    1
176  NOCACHE
177  NOCYCLE;
```

Script Output — Task completed in 0,036 seconds

```
Table DW_DATA.CLIENT_DIMENSION created.


Sequence SEQ_CLIENT_DIM created.
```

2) create table dw_data. dish_dimension and sequence seq_dish_dim

```
114 CREATE TABLE dw_data.dish_dimension(
115 dish_id NUMBER,
116 dish_name VARCHAR2(20) NOT NULL,
117 dish_category VARCHAR2(20) NOT NULL,
118 price DECIMAL (10,2) NOT NULL,
119 composition VARCHAR2(150) NOT NULL,
120 weight DECIMAL (10,2) NOT NULL,
121 status CHAR(1) NOT NULL CHECK (status IN ('N','Y')),
122 CONSTRAINT DISH_ID_PK PRIMARY KEY ( dish_id ) ENABLE);
123
```

Script Output | Task completed in 0,08 seconds

```
Table DW_DATA.DISH_DIMENSION created.
```

```
125 CREATE SEQUENCE SEQ_DISH_DIM
126  START WITH     1
127  INCREMENT BY   1
128  NOCACHE
129  NOCYCLE;
```

Script Output | Task completed in 0,051 seconds

```
Table DW_DATA.DISH_DIMENSION created.


Sequence SEQ_DISH_DIM created.
```

3) create table dw_data. employee_dimension and sequence seq_employee_dim

```
205 CREATE TABLE dw_data.employee_dimension(
206 employee_id NUMBER,
207 first_name VARCHAR2(20) NOT NULL,
208 last_name VARCHAR2(20) NOT NULL,
209 phone VARCHAR2(20) NOT NULL,
210 email VARCHAR2(40) NOT NULL,
211 department VARCHAR2(20) NOT NULL,
212 job_title VARCHAR2(20) NOT NULL,
213 address VARCHAR2(40) NOT NULL,
214 country VARCHAR2(20) NOT NULL,
215 city VARCHAR2(20) NOT NULL,
216 building INT,
217 apartment INT,
218 status CHAR(1) NOT NULL CHECK (status IN ('N','Y')),
219 CONSTRAINT EMPLOYEE_UNIQUE UNIQUE (phone, email),
220 CONSTRAINT EMPLOYEE_ID_PK PRIMARY KEY ( employee_id ) ENABLE);
```

Script Output | Task completed in 0,086 seconds

```
Table DW_DATA.EMPLOYEE_DIMENSION created.
```

```
223 CREATE SEQUENCE SEQ_EMPLOYEE_DIM
224  START WITH     1
225  INCREMENT BY   1
226  NOCACHE
227  NOCYCLE;
```

Script Output | Task completed in 0,038 seconds

```
Table DW_DATA.EMPLOYEE_DIMENSION created.


Sequence SEQ_EMPLOYEE_DIM created.
```

4) create table dw_data. dim_gen_period and sequence seq_gen_period_dim

```
249 CREATE TABLE dw_data.dim_gen_period (
250     period_id       NUMBER,
251     valid_from      DATE NOT NULL,
252     valid_to        DATE NOT NULL,
253     promotion_name  VARCHAR2(100) NOT NULL,
254     promotion_percent DECIMAL (5,2) NOT NULL,
255     decription      VARCHAR2(100) NOT NULL,
256     CONSTRAINT GEN_PERIOD_ID_PK PRIMARY KEY ( period_id) ENABLE
257 );
```

Script Output | Task completed in 0,064 seconds

```
Table DW_DATA.DIM_GEN_PERIOD created.
```

```
260 CREATE SEQUENCE SEQ_GEN_PERIOD_DIM
261   START WITH      1
262   INCREMENT BY    1
263   NOCACHE
264   NOCYCLE;
```

Script Output | Task completed in 0,04 seconds

```
Table DW_DATA.DIM_GEN_PERIOD created.


Sequence SEQ_GEN_PERIOD_DIM created.
```

5) create table dw_data. payment_method_dimension and sequence seq_payment_method_dim

```
232 CREATE TABLE dw_data.payment_method_dimension(
233 payment_method_id NUMBER,
234 payment_method_name VARCHAR2(20) NOT NULL,
235 status CHAR(1) NOT NULL CHECK (status IN ('N','Y')),
236 CONSTRAINT PAYMENT_METHOD_UNIQUE UNIQUE (payment_method_name),
237 CONSTRAINT PAYMENT_METHOD_ID_PK PRIMARY KEY ( payment_method_id ) ENABLE);
```

Script Output | Task completed in 0,067 seconds

```
Table DW_DATA.PAYMENT_METHOD_DIMENSION created.
```

```
240 CREATE SEQUENCE SEQ_PAYMENT_METHOD_DIM
241   START WITH      1
242   INCREMENT BY    1
243   NOCACHE
244   NOCYCLE;
```

Script Output | Task completed in 0,041 seconds

```
Table DW_DATA.PAYMENT_METHOD_DIMENSION created.


Sequence SEQ_PAYMENT_METHOD_DIM created.
```

6) create table dw_data. restaurant_dimension and sequence seq_restaurant_dim

```
182 CREATE TABLE dw_data.restaurant_dimension(
183 restaurant_id NUMBER,
184 phone VARCHAR2(20) NOT NULL,
185 email VARCHAR2(40) NOT NULL,
186 address VARCHAR2(40) NOT NULL,
187 country VARCHAR2(20) NOT NULL,
188 city VARCHAR2(20) NOT NULL,
189 building INT,
190 apartment INT,
191 status CHAR(1) NOT NULL CHECK (status IN ('N','Y')),
192 CONSTRAINT RESTAURANT_UNIQUE UNIQUE (phone, email),
193 CONSTRAINT RESTAURANT_ID_PK PRIMARY KEY ( restaurant_id ) ENABLE);
```

Script Output | Task completed in 0,075 seconds

```
Table DW_DATA.RESTAURANT_DIMENSION created.
```

```
196 CREATE SEQUENCE SEQ_RESTAURANT_DIM
197  START WITH     1
198  INCREMENT BY   1
199  NOCACHE
200  NOCYCLE;
```

Script Output | Task completed in 0,042 seconds

```
Table DW_DATA.RESTAURANT_DIMENSION created.


Sequence SEQ_RESTAURANT_DIM created.
```

7) create table dw_data. order_fact and sequence seq_order_fact

```
19 CREATE TABLE dw_data.order_fact(
20 order_id NUMBER,
21 client_id NUMBER NOT NULL,
22 employee_id NUMBER NOT NULL,
23 restaurant_id NUMBER NOT NULL,
24 date_id DATE NOT NULL,
25 period_id NUMBER NOT NULL,
26 payment_method_id NUMBER NOT NULL,
27 dish_id NUMBER NOT NULL,
28 dish_amount INT NOT NULL,
29 total_cost DECIMAL (11,2) NOT NULL,
30 delivery CHAR(1) NOT NULL CHECK (delivery IN ('N','Y')),
31 CONSTRAINT ORDER_ID_PK PRIMARY KEY ( order_id ) ENABLE)
32 PARTITION BY RANGE (date_id)
33 subpartition by hash(client_id) subpartitions 4
34 (
35 PARTITION quarter_1 VALUES LESS THAN(to_date('01.04.2021','DD.MM.YYYY'))
36     (
37     subpartition quarter_1_sub_1,
```

Script Output | Task completed in 0,103 seconds

```
Table DW_DATA.ORDER_FACT created.
```

```
106 CREATE SEQUENCE SEQ_ORDER_FACT
107  START WITH     1
108  INCREMENT BY   1
109  NOCACHE
110  NOCYCLE;
```

Script Output | Task completed in 0,05 seconds

```
Table DW_DATA.ORDER_FACT created.


Sequence SEQ_ORDER_FACT created.
```

Add constraints:

```
65  ALTER TABLE dw_data.order_fact
66   ADD CONSTRAINT client_fk
67       FOREIGN KEY (client_id)
68       REFERENCES dw_data.client_dimension(client_id);
69
70  ALTER TABLE dw_data.order_fact
71   ADD CONSTRAINT employee_fk
```

Script Output | Task completed in 0,322 seconds

```
Table DW_DATA.ORDER_FACT altered.


Table DW_DATA.ORDER_FACT altered.


Table DW_DATA.ORDER_FACT altered.


Table DW_DATA.ORDER_FACT altered.


Table DW_DATA.ORDER_FACT altered.


Table DW_DATA.ORDER_FACT altered.
```

Create all necessary stored procedure for moving all data from cleansing layer to DW layer. The main task of these procedures is to convert natural keys to primary keys.

1) create package pkg_etl_clients_dw and stored procedure load_clean_clients_dw

```
1  CREATE OR REPLACE PACKAGE pkg_etl_clients_dw
2   AS
3      PROCEDURE load_CLEAN_CLIENTS_DW;
4   END pkg_etl_clients_dw;
```

Script Output | Task completed in 0,163 seconds

```
Package PKG_ETL_CLIENTS_DW compiled
```

```
6  CREATE OR REPLACE PACKAGE body pkg_etl_clients_dw
7  AS
8    PROCEDURE load_CLEAN_CLIENTS_DW
9     AS
10      BEGIN
11       DECLARE
12        TYPE CURSOR_VARCHAR IS TABLE OF varchar2(100);
13        TYPE CURSOR_NUMBER IS TABLE OF number(10);
14        TYPE BIG_CURSOR IS REF CURSOR ;
15
16        ALL_INF BIG_CURSOR;
17
18        CLIENT_FIRST_NAME CURSOR_VARCHAR;
19        CLIENT_LAST_NAME CURSOR_VARCHAR;
20        CLIENT_PHONE CURSOR_VARCHAR;
21        CLIENT_EMAIL CURSOR_VARCHAR;
22        CLIENT_STREET CURSOR_VARCHAR;
23        CLIENT_COUNTRY CURSOR_VARCHAR;
24        CLIENT_CITY CURSOR_VARCHAR;
25        CLIENT_STATUS CURSOR_VARCHAR;
26        FIRST_NAME_stage CURSOR_VARCHAR;
```

Script Output | Query Result | Task completed in 0,716 seconds

```
Package Body PKG_ETL_CLIENTS_DW compiled
```

2) create package pkg_etl_dishes_dw and stored procedure load_clean_dishes_dw

```
1  CREATE OR REPLACE PACKAGE pkg_etl_dishes_dw
2  AS
3     PROCEDURE load_CLEAN_DISHES_DW;
4  END pkg_etl_dishes_dw;
```

Script Output

Task completed in 0,104 seconds

```
Package PKG_ETL_DISHES_DW compiled
```

```
8   PROCEDURE load_CLEAN_DISHES_DW
9    AS
10     BEGIN
11      DECLARE
12       TYPE CURSOR_VARCHAR IS TABLE OF varchar2(200);
13       TYPE CURSOR_NUMBER IS TABLE OF number(10);
14       TYPE CURSOR_DECIMAL IS TABLE OF decimal(10,2);
15       TYPE BIG_CURSOR IS REF CURSOR ;
16
17       ALL_INF BIG_CURSOR;
18
19       DISH_NAME CURSOR_VARCHAR;
20       DISH_CATEGORY CURSOR_VARCHAR;
21       DISH_PRICE CURSOR_DECIMAL;
22       DISH_COMPOSITION CURSOR_VARCHAR;
23       DISH_WEIGHT CURSOR_DECIMAL;
24       DISH_STATUS CURSOR_VARCHAR;
25       DISH_NAME_stage CURSOR_VARCHAR;
26       DISH_PRICE_stage CURSOR_DECIMAL;
27       DISH_ID CURSOR_NUMBER;
```

Script Output | Query Result

Task completed in 0,638 seconds

```
Package Body PKG_ETL_DISHES_DW compiled
```

3) create package pkg_etl_employees_dw and stored procedure load_clean_employees_dw

```
1  CREATE OR REPLACE PACKAGE pkg_etl_employees_dw
2  AS
3     PROCEDURE load_CLEAN_EMPLOYEES_DW;
4  END pkg_etl_employees_dw;
```

Script Output

Task completed in 0,101 seconds

```
Package PKG_ETL_EMPLOYEES_DW compiled
```

```
CREATE OR REPLACE PACKAGE body pkg_etl_employees_dw
 AS
  PROCEDURE load_CLEAN_EMPLOYEES_DW
   AS
     BEGIN
      DECLARE
       TYPE CURSOR_VARCHAR IS TABLE OF varchar2(100);
       TYPE CURSOR_NUMBER IS TABLE OF number(10);
       TYPE CURSOR_INT IS TABLE OF int;
       TYPE BIG_CURSOR IS REF CURSOR ;

       ALL_INF BIG_CURSOR;

       EMPLOYEE_FIRST_NAME CURSOR_VARCHAR;
       EMPLOYEE_LAST_NAME CURSOR_VARCHAR;
       EMPLOYEE_PHONE CURSOR_VARCHAR;
       EMPLOYEE_EMAIL CURSOR_VARCHAR;
       EMPLOYEE_DEPARTMENT CURSOR_VARCHAR;
       EMPLOYEE_JOB_TITLE CURSOR_VARCHAR;
       EMPLOYEE_STREET CURSOR_VARCHAR;
       EMPLOYEE_COUNTRY CURSOR_VARCHAR;
```

Script Output | Query Result

Task completed in 0,674 seconds

```
Package Body PKG_ETL_EMPLOYEES_DW compiled
```

4) create package pkg_etl_gen_periods_dw and stored procedure load_clean_gen_periods_dw

```
CREATE OR REPLACE PACKAGE pkg_etl_gen_periods_dw
AS
    PROCEDURE load_CLEAN_GEN_PERIODS_DW;
END pkg_etl_gen_periods_dw;
```

Script Output

Task completed in 0,386 seconds

```
Package PKG_ETL_GEN_PERIODS_DW compiled
```

```
CREATE OR REPLACE PACKAGE body pkg_etl_gen_periods_dw
AS
  PROCEDURE load_CLEAN_GEN_PERIODS_DW
   AS
     BEGIN
     MERGE INTO dw_data.dim_gen_period A
     USING ( SELECT valid_from, valid_to, promotion_name, promotion_percent, decription
             FROM dw_cl.cl_t_gen_period) B
             ON (a.valid_from=b.valid_from AND a.valid_to=b.valid_to AND a.promotion_name=b.promotion_name)
             WHEN MATCHED THEN
                UPDATE SET a.promotion_percent=b.promotion_percent,
                           a.decription=b.decription
             WHEN NOT MATCHED THEN
                INSERT (a.period_id, a.valid_from, a.valid_to, a.promotion_name, a.promotion_percent, a.decription)
                VALUES (SEQ_GEN_PERIOD_DIM.NEXTVAL, b.valid_from, b.valid_to, b.promotion_name, b.promotion_percent, b.decription);
     COMMIT;
   END load_CLEAN_GEN_PERIODS_DW;
END pkg_etl_gen_periods_dw;
```

Script Output | Query Result

Task completed in 0,772 seconds

```
Package Body PKG_ETL_GEN_PERIODS_DW compiled
```

5) create package pkg_etl_payment_methods_dw and stored procedure load_clean_payment_methods_dw

```
1  CREATE OR REPLACE PACKAGE pkg_etl_payment_methods_dw
2  AS
3      PROCEDURE load_CLEAN_PAYMENT_METHODS_DW;
4  END pkg_etl_payment_methods_dw;
```

Script Output | Task completed in 0,129 seconds

```
Package PKG_ETL_PAYMENT_METHODS_DW compiled
```

```
6  CREATE OR REPLACE PACKAGE body pkg_etl_payment_methods_dw
7  AS
8    PROCEDURE load_CLEAN_PAYMENT_METHODS_DW
9     AS
10      BEGIN
11      MERGE INTO dw_data.payment_method_dimension A
12      USING ( SELECT payment_method_name, payment_method_status
13              FROM dw_cl.cl_t_payment_method) B
14              ON (a.payment_method_name=b.payment_method_name)
15              WHEN MATCHED THEN
16                 UPDATE SET a.status=b.payment_method_status
17              WHEN NOT MATCHED THEN
18                 INSERT (a.payment_method_id, a.payment_method_name, a.status)
19                 VALUES (SEQ_PAYMENT_METHOD_DIM.NEXTVAL, b.payment_method_name, b.payment_method_status);
20      COMMIT;
21    END load_CLEAN_PAYMENT_METHODS_DW;
22  END pkg_etl_payment_methods_dw;
23
```

Script Output | Query Result | Task completed in 0,575 seconds

```
Package Body PKG_ETL_PAYMENT_METHODS_DW compiled
```

6) create package pkg_etl_restaurants_dw and stored procedure load_clean_restaurants_dw

```
1  CREATE OR REPLACE PACKAGE pkg_etl_restaurants_dw
2  AS
3      PROCEDURE load_CLEAN_RESTAURANTS_DW;
4  END pkg_etl_restaurants_dw;
```

Script Output | Task completed in 0,098 seconds

```
Package PKG_ETL_RESTAURANTS_DW compiled
```

```
6  CREATE OR REPLACE PACKAGE body pkg_etl_restaurants_dw
7  AS
8    PROCEDURE load_CLEAN_RESTAURANTS_DW
9     AS
10      BEGIN
11       DECLARE
12        TYPE CURSOR_VARCHAR IS TABLE OF varchar2(100);
13        TYPE CURSOR_NUMBER IS TABLE OF number(10);
14        TYPE CURSOR_INT IS TABLE OF int;
15        TYPE BIG_CURSOR IS REF CURSOR ;
16
17        ALL_INF BIG_CURSOR;
18
19        RESTAURANT_PHONE CURSOR_VARCHAR;
20        RESTAURANT_EMAIL CURSOR_VARCHAR;
21        RESTAURANT_STREET CURSOR_VARCHAR;
22        RESTAURANT_COUNTRY CURSOR_VARCHAR;
23        RESTAURANT_CITY CURSOR_VARCHAR;
24        RESTAURANT_BUILDING CURSOR_INT;
25        RESTAURANT_APARTMENT CURSOR_INT;
```

Script Output | Query Result | Task completed in 0,659 seconds

```
Package Body PKG_ETL_RESTAURANTS_DW compiled
```

7) create pkg_etl_transactions_dw and stored procedure load_clean_transactions_dw

```
1  CREATE OR REPLACE PACKAGE pkg_etl_transactions_dw
2  AS
3     PROCEDURE load_CLEAN_TRANSACTIONS_DW;
4  END pkg_etl_transactions_dw;
```

Script Output | Task completed in 0,113 seconds

```
Package PKG_ETL_TRANSACTIONS_DW compiled
```

```
6  CREATE OR REPLACE PACKAGE body pkg_etl_transactions_dw
7  AS
8    PROCEDURE load_CLEAN_TRANSACTIONS_DW
9     AS
10      BEGIN
11       DECLARE
12        period_id_v NUMBER;
13        TYPE CURSOR_VARCHAR IS TABLE OF varchar2(100);
14        TYPE CURSOR_NUMBER IS TABLE OF number(10);
15        TYPE CURSOR_DATE IS TABLE OF date;
16        TYPE CURSOR_DECIMAL IS TABLE OF decimal(11,2);
17        TYPE BIG_CURSOR IS REF CURSOR ;
18
19        ALL_INF BIG_CURSOR;
20
21        CLIENT_ID CURSOR_NUMBER;
22        EMPLOYEE_ID CURSOR_NUMBER;
23        RESTAURANT_ID CURSOR_NUMBER;
24        DATE_ID CURSOR_DATE;
25        PAYMENT_METHOD_ID CURSOR_NUMBER;
```

Script Output | Query Result | Task completed in 0,67 seconds

```
Package Body PKG_ETL_TRANSACTIONS_DW compiled
```

Run all procedures:

```
1  BEGIN
2      pkg_etl_clients_dw.load_CLEAN_CLIENTS_DW;
3      pkg_etl_restaurants_dw.load_CLEAN_RESTAURANTS_DW;
4      pkg_etl_employees_dw.load_CLEAN_EMPLOYEES_DW;
5      pkg_etl_dishes_dw.load_CLEAN_DISHES_DW;
6      pkg_etl_payment_methods_dw.load_CLEAN_PAYMENT_METHODS_DW;
7      pkg_etl_gen_periods_dw.load_CLEAN_GEN_PERIODS_DW;
8      pkg_etl_transactions_dw.load_CLEAN_TRANSACTIONS_DW;
9  END;
```

Script Output | Task completed in 203,964 seconds

```
PL/SQL procedure successfully completed.
```

Select data from tables on DW layer:

1) dw_data.client_dimension

```
11  SELECT * FROM  dw_data.client_dimension;
```

Query Result | SQL | All Rows Fetched: 5 in 0,035 seconds

	CLIENT_ID	FIRST_NAME	LAST_NAME	PHONE	EMAIL	STREET	COUNTRY	CITY	STATUS
1	1	Adriana	Karnitskaya	375294217925	AdrianaKarnitskaya@mail.ru	Yesenin	Belarus	Minsk	Y
2	2	Ian	Astafyev	375295321165	IanAstafyev@mail.ru	Champs Elysees	France	Paris	Y
3	3	Ada	Alymova	375292376362	AdaAlymova@mail.ru	Nikolskaya	Russia	Moscow	Y
4	4	Pavel	Moshko	375291330060	PavelMoshko@mail.ru	Antoine Dansaert	Belgium	Brussels	Y
5	5	Alice	Lysenko	375294487084	AliceLysenko@mail.ru	Fifth Avenue	USA	New York	Y

2) dw_data. restaurant_dimension

```
13  SELECT * FROM dw_data.restaurant_dimension;
```

Query Result | SQL | All Rows Fetched: 5 in 0,137 seconds

	RESTAURANT_ID	PHONE	EMAIL	ADDRESS	COUNTRY	CITY	BUILDING	APARTMENT	STATUS
1	1	375295841669	5@mail.ru	Gurchevskaya	Poland	Warsaw	46	230	Y
2	2	375292356572	3@mail.ru	Fifth Avenue	USA	New York	30	189	Y
3	3	375299013465	2@mail.ru	Nikolskaya	Russia	Moscow	32	165	Y
4	4	375295176493	4@mail.ru	Vladimirskaya	Ukraine	Kiev	36	185	Y
5	5	375294308850	1@mail.ru	Yesenin	Belarus	Minsk	81	132	Y

3) dw_data. employee_dimension

```
15  SELECT * FROM dw_data.employee_dimension;
```

Query Result | SQL | All Rows Fetched: 25 in 0,034 seconds

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	PHONE	EMAIL	DEPARTMENT	JOB_TITLE	ADDRESS	COUN
1	1	Adriana	Karnitskaya	375291698908	AdrianaKarnitskaya@mail.ru	department_name_1	director	Yesenin	Belaru
2	2	Alyssa	Malysheva	375290801452	AlyssaMalysheva@mail.ru	department_name_4	employee	Vladimirskaya	Ukrai
3	3	Rita	Astafyeva	375299300168	RitaAstafyeva@mail.ru	department_name_2	manager	K Kaiser-Friedrich	Germa
4	4	Clara	Zaykova	375295778867	ClaraZaykova@mail.ru	department_name_2	manager	Abby	Engla
5	5	Eva	Moshko	375292569304	EvaMoshko@mail.ru	department_name_4	employee	Aloyas	Latvi
6	6	Ivan	Semin	375296331501	IvanSemin@mail.ru	department_name_4	employee	Via del Corso	Belgi
7	7	Ian	Astafyev	375297942988	IanAstafyev@mail.ru	department_name_3	employee	Champs Elysees	Franc
8	8	Ada	Alymova	375299826609	AdaAlymova@mail.ru	department_name_2	manager	Nikolskaya	Russi
9	9	Maxim	Mayorov	375291285449	MaximMayorov@mail.ru	department_name_2	manager	D Dunkri	Engla
10	10	Veronika	Sadovskaya	375296747159	VeronikaSadovskaya@mail.ru	department_name_3	employee	D Dunkri	Eston
11	11	Pavel	Moshko	375296156608	PavelMoshko@mail.ru	department_name_4	employee	Antoine Dansaert	Belgi
12	12	Alexandra	Etkina	375295186852	AlexandraEtkina@mail.ru	department_name_1	director	Graben	Austr
13	13	Nikita	Malyshev	375291493172	NikitaMalyshev@mail.ru	department_name_4	employee	G Galve	Latvi
14	14	Miron	Parfenov	375292324435	MironParfenov@mail.ru	department_name_3	employee	Antoine Dansaert	Franc
15	15	Oleg	Etkin	375291680917	OlegEtkin@mail.ru	department_name_1	director	Abby	Belar

4) dw_data. dish_dimension

```
17  SELECT * FROM dw_data.dish_dimension;
```

Query Result | SQL | All Rows Fetched: 5 in 0,042 seconds

	DISH_ID	DISH_NAME	DISH_CATEGORY	PRICE	COMPOSITION	WEIGHT	STATUS
1	1	soup	hot	15	soup ingredients	915	Y
2	2	pizza	hot	25	pizza ingredients	486	Y
3	3	pasta	hot	87	pasta ingredients	959	Y
4	4	chebupelli	hot	61	chebupelli ingredients	512	Y
5	5	greek salad	appetizer	82	greek salad ingredients	797	Y

5) dw_data. payment_method_dimension

```
19  SELECT * FROM dw_data.payment_method_dimension;
```

Query Result | SQL | All Rows Fetched: 2 in 0,051 seconds

	PAYMENT_METHOD_ID	PAYMENT_METHOD_NAME	STATUS
1	1	bank card	Y
2	2	cash	Y

6) dw_data. dim_gen_period

```
21  SELECT * FROM dw_data.dim_gen_period;
```

Query Result | SQL | All Rows Fetched: 6 in 0,03 seconds

	PERIOD_ID	VALID_FROM	VALID_TO	PROMOTION_NAME	PROMOTION_PERCENT	DECRIPTION
1	1	01.04.21	30.04.21	promotion_name_4	5	decription_4
2	2	01.02.21	28.02.21	promotion_name_2	15	decription_2
3	3	01.06.21	31.12.21	promotion_name_6	20	decription_6
4	4	01.03.21	31.03.21	promotion_name_3	10	decription_3
5	5	01.05.21	31.05.21	promotion_name_5	20	decription_5
6	6	01.01.21	31.01.21	promotion_name_1	25	decription_1

7) dw_data. order_fact

```
23  SELECT * FROM dw_data.order_fact
24  ORDER BY 1;
```

Script Output | Query Result | SQL | Fetched 50 rows in 0,163 seconds

	ORDER_ID	CLIENT_ID	EMPLOYEE_ID	RESTAURANT_ID	DATE_ID	PERIOD_ID	PAYMENT_METHOD_ID	DISH_ID	DISH_AMOUNT	TOTAL_COST	DELIVERY
1	1	3	6	5	03.01.21	6	2	2	47	1179	Y
2	2	2	18	1	04.01.21	6	1	2	38	963	Y
3	3	3	16	3	04.01.21	6	2	4	3	187	Y
4	4	4	16	3	04.01.21	6	2	2	25	646	Y
5	5	5	6	1	03.01.21	6	2	3	12	1048	Y
6	6	4	12	2	04.01.21	6	2	1	91	1378	Y
7	7	3	3	2	04.01.21	6	2	5	10	867	Y
8	8	1	3	4	04.01.21	6	2	1	84	1263	Y
9	9	4	3	1	04.01.21	6	2	3	1	136	Y
10	10	4	6	2	04.01.21	6	2	5	5	442	Y
11	11	3	18	2	04.01.21	6	2	5	14	1228	Y
12	12	2	18	2	04.01.21	6	2	2	56	1412	Y
13	13	4	16	5	05.01.21	6	1	5	2	202	Y
14	14	3	12	1	05.01.21	6	1	4	21	1318	Y