Таблица Delivery

|  |  |  |
| --- | --- | --- |
| Столбец | Тип | Назначение |
| DELIVERY\_ID | INT | PK, по которому будут определяется уникальная доставка |
| ADRESS | VARCHAR(200) | Адрес |
| PAYMENT\_METHOD | VARCHAR(100) | Метод оплаты |
| DELIVERY\_METHOD | VARCHAR(100) | Метод доставки |
| DATE\_OF\_DELIVERY | VARCHAR(100) | Дата, на которую назначена доставки |

Таблица ORDERS

|  |  |  |
| --- | --- | --- |
| Столбец | Тип | Назначение |
| ORDER\_ID | INT | PK, по которому определяется уникальный заказ |
| ORDER\_DATE | VARCHAR(100) | Дата заказа |
| ORDER\_STATUS | VARCHAR(100) | Статус заказа |
| DELIVERY\_ID | INT | FK, который содержит ссылку на таблицу Delivery |
| USER\_ID | INT | FK, который содержит ссылку на таблицу DB\_Users |

Таблица DB\_USERS

|  |  |  |
| --- | --- | --- |
| Столбец | Тип | Назначение |
| USER\_ID | INT | PK, по которому определяется уникальный пользователь |
| LOGIN | VARCHAR(100) | Логин пользователя |
| EMAIL | VARCHAR(100) | Email пользователя |
| PASSWORD | VARCHAR(100) | Пароль пользователя |

Таблица CART

|  |  |  |
| --- | --- | --- |
| Столбец | Тип | Назначение |
| CART\_ID | INT | PK, по которому определяется уникальная корзина пользователя |
| USER\_ID | INT | FK, который содержит ссылку на пользователя |
| QUANTITY | INT | Количество товаров в корзине |

Таблица ITEM\_IN\_CART

|  |  |  |
| --- | --- | --- |
| Столбец | Тип | Назначение |
| ITEM\_IN\_CART\_ID | INT | PK, по которому определяется уникальная запись связи таблиц CART и PRODUCT |
| CART\_ID | INT | FK, который содержит ссылку на пользователя |
| PRODUCT\_ID | INT | FK, который содержит ссылку на продукт |

Таблица PRODUCT

|  |  |  |
| --- | --- | --- |
| Столбец | Тип | Назначение |
| PRODUCT\_ID | INT | PK, по которому определяется уникальная запись в таблице PRODUCT |
| PRODUCT\_NAME | VARCHAR(100) | Имя вещи |
| PRODUCT\_MATERIAL | VARCHAR(100) | Материал |
| PRODUCT\_SIZE | VARCHAR(100) | Размер в м^2 |
| PRODUCT\_TYPE | VARCHAR(100) | Тип вещи |
| STORAGE\_ID | INT | FK, который содержит ссылку на склад |

ТАБЛИЦА STORAGE

|  |  |  |
| --- | --- | --- |
| Столбец | Тип | Назначение |
| STORAGE\_ID | INT | PK, по которому определяется уникальный склад |
| ADRESS | VARCHAR(100) | Адрес склада |
| COUNTRY | VARCHAR(100) | Страна, в которой располагается склад |
| CAPACITY | INT | Вместимость склад |

ORACLE

Таблица Delivery

|  |  |  |
| --- | --- | --- |
| Столбец | Тип | Назначение |
| DELIVERY\_ID | NUMBER | PK, по которому будут определяется уникальная доставка |
| ADRESS | VARCHAR2(200) | Адрес |
| PAYMENT\_METHOD | VARCHAR2(100) | Метод оплаты |
| DELIVERY\_METHOD | VARCHAR2(100) | Метод доставки |
| DATE\_OF\_DELIVERY | VARCHAR2(100) | Дата, на которую назначена доставки |

Таблица ORDERS

|  |  |  |
| --- | --- | --- |
| Столбец | Тип | Назначение |
| ORDER\_ID | NUMBER | PK, по которому определяется уникальный заказ |
| ORDER\_DATE | VARCHAR2(100) | Дата заказа |
| ORDER\_STATUS | VARCHAR2(100) | Статус заказа |
| DELIVERY\_ID | NUMBER | FK, который содержит ссылку на таблицу Delivery |
| USER\_ID | NUMBER | FK, который содержит ссылку на таблицу DB\_Users |

Таблица DB\_USERS

|  |  |  |
| --- | --- | --- |
| Столбец | Тип | Назначение |
| USER\_ID | NUMBER | PK, по которому определяется уникальный пользователь |
| LOGIN | VARCHAR2(100) | Логин пользователя |
| EMAIL | VARCHAR2(100) | Email пользователя |
| PASSWORD | VARCHAR2(100) | Пароль пользователя |

Таблица CART

|  |  |  |
| --- | --- | --- |
| Столбец | Тип | Назначение |
| CART\_ID | NUMBER | PK, по которому определяется уникальная корзина пользователя |
| USER\_ID | NUMBER | FK, который содержит ссылку на пользователя |
| QUANTITY | NUMBER | Количество товаров в корзине |

Таблица ITEM\_IN\_CART

|  |  |  |
| --- | --- | --- |
| Столбец | Тип | Назначение |
| ITEM\_IN\_CART\_ID | NUMBER | PK, по которому определяется уникальная запись связи таблиц CART и PRODUCT |
| CART\_ID | NUMBER | FK, который содержит ссылку на пользователя |
| PRODUCT\_ID | NUMBER | FK, который содержит ссылку на продукт |

Таблица PRODUCT

|  |  |  |
| --- | --- | --- |
| Столбец | Тип | Назначение |
| PRODUCT\_ID | NUMBER | PK, по которому определяется уникальная запись в таблице PRODUCT |
| PRODUCT\_NAME | VARCHAR2(100) | Имя вещи |
| PRODUCT\_MATERIAL | VARCHAR2(100) | Материал |
| PRODUCT\_SIZE | VARCHAR2(100) | Размер в м^2 |
| PRODUCT\_TYPE | VARCHAR2(100) | Тип вещи |
| STORAGE\_ID | NUMBER | FK, который содержит ссылку на склад |

ТАБЛИЦА STORAGE

|  |  |  |
| --- | --- | --- |
| Столбец | Тип | Назначение |
| STORAGE\_ID | NUMBER | PK, по которому определяется уникальный склад |
| ADRESS | VARCHAR2(100) | Адрес склада |
| COUNTRY | VARCHAR2(100) | Страна, в которой располагается склад |
| CAPACITY | NUMBER | Вместимость склад |

|  |  |  |
| --- | --- | --- |
| Название | Тип объекта | Назначение |
| INFORMATION\_ABOUT\_ORDER | VIEW | Получение полных данных о заказе, включая прошедшие статусы заказа, информацию о клиенте. |
| INSERT\_USER | PROCEDURE | Добавление нового пользователя в бд |
| GET\_USERS | FUNCTION | Получение всех зарегистрированных пользователей |
| INSERT\_PRODUCT\_TO\_CART | PROCEDURE | Добавление нового продукта в корзину |
| GET\_CART\_INFO\_BY\_ID | FUNCTION | Получение корзины пользователя по ID |
| INSERT\_ITEM\_IN\_CART | PROCEDURE | Добавление данных в таблицу ITEM\_IN\_CART, которая служит для реализации связи many-to-many таблиц CART И PRODUCT |
| GET\_ITEMS\_IN\_CART | FUNCTION | Получение данных из таблицы ITEM\_IN\_CART |
| INSERT\_PRODUCT | PROCEDURE | Добавление нового продукта на склад |
| INSERT\_DELIVERY | PROCEDURE | Добавление новой записи в таблицу DELIVERY |
| INSERT\_ORDERS | PROCEDURE | Добавление новой записи в таблицу ORDERS |
| HASH\_PASSWORD | TRIGGER | Хэширует пароль пользователя |

Листинг кода на MSSQL

|  |
| --- |
| ---------------- CREATING TABLES ---------------------  CREATE TABLE DB\_USERS(  USER\_ID INT PRIMARY KEY,  LOGIN VARCHAR(100),  EMAIL VARCHAR(100),  PASSWORD VARCHAR(100)  );  CREATE TABLE CART(  CART\_ID INT PRIMARY KEY,  USER\_ID INT UNIQUE,  QUANTITY INT,  CONSTRAINT USER\_ID\_CONSTRAINT FOREIGN KEY (USER\_ID) REFERENCES DB\_USERS(USER\_ID)  );  DROP TABLE CART;  DROP TABLE DB\_USERS;  DROP TABLE ITEM\_IN\_CART;  CREATE TABLE ITEM\_IN\_CART(  ITEM\_IN\_CART\_ID INT PRIMARY KEY,  CART\_ID INT,  PRODUCT\_ID INT,  CONSTRAINT CART\_ID\_CONSTRAINT FOREIGN KEY (CART\_ID) REFERENCES CART(CART\_ID),  CONSTRAINT PRODUCT\_ID\_CONSTRAINT FOREIGN KEY (PRODUCT\_ID) REFERENCES PRODUCT(PRODUCT\_ID),  )  CREATE TABLE PRODUCT(  PRODUCT\_ID INT PRIMARY KEY,  PRODUCT\_NAME VARCHAR(100),  PRODUCT\_MATERIAL VARCHAR(100),  PRODUCT\_SIZE VARCHAR(100),  PRODUCT\_TYPE VARCHAR(100),  STORAGE\_ID INT,  CONSTRAINT STORAGE\_ID\_CONSTRAINT FOREIGN KEY (STORAGE\_ID) REFERENCES STORAGE(STORAGE\_ID)  );  CREATE TABLE STORAGE(  STORAGE\_ID INT PRIMARY KEY,  ADRESS VARCHAR(100),  COUNTRY VARCHAR(100),  CAPACITY INT  );  CREATE TABLE DELIVERY(  DELIVERY\_ID INT PRIMARY KEY,  ADRESS VARCHAR(200),  PAYMENT\_METHOD VARCHAR(100),  DELIVERY\_METHOD VARCHAR(100),  DATE\_OF\_DELIVERY VARCHAR(100)  );  CREATE TABLE ORDERS(  ORDER\_ID INT PRIMARY KEY,  ORDER\_DATE VARCHAR(100),  ORDER\_STATUS VARCHAR(100),  DELIVERY\_ID INT,  USER\_ID INT,  CONSTRAINT USER\_ID\_CONSTRAIN FOREIGN KEY (USER\_ID) REFERENCES DB\_USERS(USER\_ID),  CONSTRAINT DELIVERY\_ID\_CONSTRAIN FOREIGN KEY (DELIVERY\_ID) REFERENCES DELIVERY(DELIVERY\_ID)  );  DROP TABLE ORDERS;  ---------------- DB\_USERS TABLE ---------------------  GO  CREATE TRIGGER HASH\_PASSWORD  ON DB\_USERS  AFTER INSERT, UPDATE  AS  BEGIN  SET NOCOUNT ON;    UPDATE DB\_USERS  SET DB\_USERS.PASSWORD = HASHBYTES('SHA2\_256', DB\_USERS.PASSWORD)  FROM inserted  WHERE DB\_USERS.USER\_ID = inserted.USER\_ID;  END;  GO  CREATE PROCEDURE INSERT\_USER  @userId INT,  @login VARCHAR(100),  @email VARCHAR(100),  @password VARCHAR(100)  AS  BEGIN  INSERT INTO DB\_USERS (USER\_ID, LOGIN, EMAIL, PASSWORD)  VALUES (@userId, @login, @email, @password)  END  EXEC INSERT\_USER  @userId = 4,  @login = 'test\_user4',  @email = 'example@example.com',  @password = 'password123';  select \*from DB\_USERS;  GO  CREATE FUNCTION GET\_USERS()  RETURNS TABLE  AS  RETURN  SELECT USER\_ID, LOGIN, EMAIL, PASSWORD  FROM DB\_USERS;  SELECT \*  FROM GET\_USERS();  ---------------- CART TABLE ---------------------  GO  CREATE PROCEDURE INSERT\_PRODUCT\_TO\_CART  @CART\_ID INT,  @USER\_ID INT,  @QUANTITY INT  AS  BEGIN  INSERT INTO CART VALUES(@CART\_ID,@USER\_ID,@QUANTITY);  END  EXEC INSERT\_PRODUCT\_TO\_CART  2,3,80;  SELECT\* FROM CART;  GO  CREATE FUNCTION GET\_CART\_INFO\_BY\_ID(  @CART\_ID INT  )  RETURNS TABLE  AS  RETURN  SELECT \*  FROM CART WHERE CART.CART\_ID = @CART\_ID;  SELECT \* FROM GET\_CART\_INFO\_BY\_ID(1);  ---------------- ITEM\_IN\_CART TABLE ---------------------  GO  CREATE PROCEDURE INSERT\_ITEM\_IN\_CART  @ITEM\_IN\_CART\_ID INT,  @CART\_ID INT,  @PRODUCT\_ID INT  AS  BEGIN  INSERT INTO ITEM\_IN\_CART VALUES(@ITEM\_IN\_CART\_ID,@CART\_ID,@PRODUCT\_ID);  END  EXEC INSERT\_ITEM\_IN\_CART 3,1,3;  GO  CREATE FUNCTION GET\_ITEMS\_IN\_CART()  RETURNS TABLE  AS  RETURN  SELECT \*  FROM ITEM\_IN\_CART;  SELECT \*  FROM GET\_ITEMS\_IN\_CART();  ---------------- PRODUCT TABLE ---------------------  GO  CREATE PROCEDURE INSERT\_PRODUCT  @PRODUCT\_ID INT,  @PRODUCT\_NAME VARCHAR(100),  @PRODUCT\_MATERIAL VARCHAR(100),  @PRODUCT\_SIZE VARCHAR(100),  @PRODUCT\_TYPE VARCHAR(100),  @STORAGE\_ID INT  AS  BEGIN  INSERT INTO PRODUCT VALUES(@PRODUCT\_ID,@PRODUCT\_NAME,@PRODUCT\_MATERIAL,@PRODUCT\_SIZE,@PRODUCT\_TYPE,@STORAGE\_ID INT);  END  EXEC INSERT\_PRODUCT 3,'TESTNAME3','TESTMATERIAL3','1M^2','ПОЛКА';  SELECT\* FROM PRODUCT;  ---------------- DELIVERY TABLE ---------------------  GO  CREATE PROCEDURE INSERT\_DELIVERY  @DELIVERY\_ID INT,  @ADRESS VARCHAR(200),  @PAYMENT\_METHOD VARCHAR(100),  @DELIVERY\_METHOD VARCHAR(100),  @DATE\_OF\_DELIVERY VARCHAR(100)  AS  BEGIN  INSERT INTO DELIVERY VALUES(@DELIVERY\_ID,@ADRESS,@PAYMENT\_METHOD,@DELIVERY\_METHOD,@DATE\_OF\_DELIVERY)  END  EXEC INSERT\_DELIVERY 3,'TESTADRESS3','CASH','КУРЬЕР','2024-04-22';  SELECT\* FROM DELIVERY;  ---------------- ORDERS TAB`LE ---------------------  CREATE PROCEDURE INSERT\_ORDERS  @ORDER\_ID INT,  @ORDER\_DATE VARCHAR(100),  @ORDER\_STATUS VARCHAR(100),  @DELIVERY\_ID INT,  @USER\_ID INT  AS  BEGIN  INSERT INTO ORDERS VALUES(@ORDER\_ID,@ORDER\_DATE,@ORDER\_STATUS,@DELIVERY\_ID,@USER\_ID);  END  EXEC INSERT\_ORDERS 4,'2024-02-30','ДОСТАВЛЕНО',1,1;  SELECT \* FROM DB\_USERS;  SELECT \*FROM DELIVERY;  select \*from ORDERS;  ---------------- VIEWS ---------------------  CREATE VIEW INFORMATION\_ABOUT\_ORDER  AS SELECT ORDERS.ORDER\_ID,ORDERS.ORDER\_DATE,ORDERS.ORDER\_STATUS,DELIVERY.ADRESS,DELIVERY.PAYMENT\_METHOD,DELIVERY.DELIVERY\_METHOD,DELIVERY.DATE\_OF\_DELIVERY,DB\_USERS.LOGIN,DB\_USERS.EMAIL,DB\_USERS.PASSWORD FROM ORDERS  INNER JOIN DELIVERY  ON DELIVERY.DELIVERY\_ID = ORDERS.DELIVERY\_ID  INNER JOIN DB\_USERS  ON ORDERS.USER\_ID=DB\_USERS.USER\_ID ;  SELECT \* FROM INFORMATION\_ABOUT\_ORDER; |

Листинг кода на ORACLE

|  |
| --- |
| -- Creating tables  CREATE TABLE DB\_USERS(  USER\_ID NUMBER PRIMARY KEY,  LOGIN VARCHAR2(100),  EMAIL VARCHAR2(100),  PASSWORD VARCHAR2(100)  );  GRANT CREATE TABLE TO admin;  SELECT username FROM dba\_users;  CREATE TABLE CART(  CART\_ID NUMBER PRIMARY KEY,  USER\_ID NUMBER UNIQUE,  QUANTITY NUMBER,  CONSTRAINT USER\_ID\_CONSTRAINT FOREIGN KEY (USER\_ID) REFERENCES DB\_USERS(USER\_ID)  );  drop table cart  drop table ITEM\_IN\_CART;  CREATE TABLE ITEM\_IN\_CART(  ITEM\_IN\_CART\_ID NUMBER PRIMARY KEY,  CART\_ID NUMBER,  PRODUCT\_ID NUMBER,  CONSTRAINT CART\_ID\_CONSTRAINT FOREIGN KEY (CART\_ID) REFERENCES CART(CART\_ID),  CONSTRAINT PRODUCT\_ID\_CONSTRAINT FOREIGN KEY (PRODUCT\_ID) REFERENCES PRODUCT(PRODUCT\_ID)  );  DROP TABLE ITEM\_IN\_CART;  DROP TABLE PRODUCT;  CREATE TABLE PRODUCT(  PRODUCT\_ID NUMBER PRIMARY KEY,  PRODUCT\_NAME VARCHAR2(100),  PRODUCT\_MATERIAL VARCHAR2(100),  PRODUCT\_SIZE VARCHAR2(100),  PRODUCT\_TYPE VARCHAR2(100),  STORAGE\_ID INT,  CONSTRAINT STORAGE\_ID\_CONSTRAINT FOREIGN KEY (STORAGE\_ID) REFERENCES STORAGE(STORAGE\_ID)  );  CREATE TABLE STORAGE(  STORAGE\_ID NUMBER PRIMARY KEY,  ADRESS VARCHAR2(100),  COUNTRY VARCHAR2(100),  CAPACITY NUMBER  );`  END;  /  ---------------- DB\_USERS TABLE ---------------------  CREATE OR REPLACE PROCEDURE INSERT\_USER(  USER\_ID IN NUMBER,  LOGIN IN VARCHAR2(100),  EMAIL IN VARCHAR2(100),  PASSWORD IN VARCHAR2(100)  )  AS  BEGIN  INSERT INTO CART (USER\_ID, LOGIN, EMAIL, PASSWORD) VALUES (USER\_ID, LOGIN, EMAIL, PASSWORD);  COMMIT;  END;  /  CREATE OR REPLACE TRIGGER HASH\_PASSWORD  BEFORE INSERT OR UPDATE ON DB\_USERS  FOR EACH ROW  BEGIN  :NEW.PASSWORD := DBMS\_CRYPTO.HASH(UTL\_RAW.CAST\_TO\_RAW(:NEW.PASSWORD), 2);  END;  /  BEGIN  INSERT\_USER(1, 'test\_user1', 'example@example.com', 'password123');  COMMIT;  END;  /  select \*from DB\_USERS;  CREATE OR REPLACE FUNCTION GET\_USERS  RETURN SYS\_REFCURSOR  AS  users\_cursor SYS\_REFCURSOR;  BEGIN  OPEN users\_cursor FOR  SELECT USER\_ID, LOGIN, EMAIL, PASSWORD  FROM DB\_USERS;    RETURN users\_cursor;  END;  DECLARE  users\_cursor SYS\_REFCURSOR;  BEGIN  users\_cursor := GET\_USERS();    DBMS\_SQL.RETURN\_RESULT(users\_cursor);  END;  /  ---------------- CART TABLE ---------------------  CREATE OR REPLACE PROCEDURE INSERT\_PRODUCT\_TO\_CART(  cart\_id IN NUMBER,  user\_id IN NUMBER,  quantity IN NUMBER  )  AS  BEGIN  INSERT INTO CART VALUES (cart\_id, user\_id, quantity);  END;  /  BEGIN  INSERT\_PRODUCT\_TO\_CART(2, 1, 180);  COMMIT;  END;  /  CREATE OR REPLACE FUNCTION GET\_CART\_INFO\_BY\_ID(  cart\_id IN NUMBER  )  RETURN SYS\_REFCURSOR  AS  cart\_cursor SYS\_REFCURSOR;  BEGIN  OPEN cart\_cursor FOR  SELECT \*  FROM CART  WHERE CART.CART\_ID = cart\_id  FETCH FIRST 1 ROW ONLY;    RETURN cart\_cursor;  END;  /  DECLARE  cart\_cursor SYS\_REFCURSOR;  BEGIN  cart\_cursor := GET\_CART\_INFO\_BY\_ID(1);    DBMS\_SQL.RETURN\_RESULT(cart\_cursor);  END;  /  ---------------- ITEM\_IN\_CART TABLE ---------------------  CREATE OR REPLACE PROCEDURE INSERT\_ITEM\_IN\_CART(  item\_in\_cart\_id IN NUMBER,  cart\_id IN NUMBER,  product\_id IN NUMBER  )  AS  BEGIN  INSERT INTO ITEM\_IN\_CART VALUES (item\_in\_cart\_id, cart\_id, product\_id);  END;  /  BEGIN  INSERT\_ITEM\_IN\_CART(2, 1, 1);  COMMIT;  END;  CREATE OR REPLACE FUNCTION GET\_ITEMS\_IN\_CART  RETURN SYS\_REFCURSOR  AS  items\_cursor SYS\_REFCURSOR;  BEGIN  OPEN items\_cursor FOR  SELECT \*  FROM ITEM\_IN\_CART;    RETURN items\_cursor;  END;  /  DECLARE  items\_cursor SYS\_REFCURSOR;  BEGIN  items\_cursor := GET\_ITEMS\_IN\_CART();    DBMS\_SQL.RETURN\_RESULT(items\_cursor);  END;  /  ---------------- PRODUCT TABLE ---------------------  CREATE OR REPLACE PROCEDURE INSERT\_PRODUCT(  product\_id IN NUMBER,  product\_name IN VARCHAR2,  product\_material IN VARCHAR2,  product\_size IN VARCHAR2,  product\_type IN VARCHAR2  )  AS  BEGIN  INSERT INTO PRODUCT VALUES (product\_id, product\_name, product\_material, product\_size, product\_type);  END;  /  -- Вызов хранимой процедуры INSERT\_PRODUCT  BEGIN  INSERT\_PRODUCT(2, 'TESTNAME2', 'TESTMATERIAL3', '10M^2', 'ШКАФ');  COMMIT;  END;  /  SELECT \* FROM PRODUCT;  ---------------- DELIVERY TABLE ---------------------  CREATE OR REPLACE PROCEDURE INSERT\_DELIVERY(  delivery\_id IN NUMBER,  address IN VARCHAR2,  payment\_method IN VARCHAR2,  delivery\_method IN VARCHAR2,  date\_of\_delivery IN VARCHAR2  )  AS  BEGIN  INSERT INTO DELIVERY VALUES (delivery\_id, address, payment\_method, delivery\_method, date\_of\_delivery);  END;  /  BEGIN  INSERT\_DELIVERY(2, 'TESTADRESS3', 'CARD', 'КУРЬЕР', '2024-04-29');  COMMIT;  END;  /  SELECT\* FROM DELIVERY;  ---------------- ORDERS TABLE ---------------------  CREATE OR REPLACE PROCEDURE INSERT\_ORDERS(  order\_id IN NUMBER,  order\_date IN VARCHAR2,  order\_status IN VARCHAR2,  delivery\_id IN NUMBER,  user\_id IN NUMBER  )  AS  BEGIN  INSERT INTO ORDERS VALUES (order\_id, order\_date, order\_status, delivery\_id, user\_id);  END;  /  BEGIN  INSERT\_ORDERS(5, '2024-04-29', 'ДОСТАВЛЕНО', 1, 2);  COMMIT;  END;  /  SELECT \* FROM DB\_USERS;  SELECT \*FROM DELIVERY;  select \*from ORDERS;  ---------------- VIEWS ---------------------  CREATE OR REPLACE VIEW INFORMATION\_ABOUT\_ORDER AS  SELECT  ORDERS.ORDER\_ID,  ORDERS.ORDER\_DATE,  ORDERS.ORDER\_STATUS,  DELIVERY.ADRESS,  DELIVERY.PAYMENT\_METHOD,  DELIVERY.DELIVERY\_METHOD,  DELIVERY.DATE\_OF\_DELIVERY,  DB\_USERS.LOGIN,  DB\_USERS.EMAIL,  DB\_USERS.PASSWORD  FROM ORDERS  INNER JOIN DELIVERY ON DELIVERY.DELIVERY\_ID = ORDERS.DELIVERY\_ID  INNER JOIN DB\_USERS ON ORDERS.USER\_ID = DB\_USERS.USER\_ID;  SELECT \* FROM INFORMATION\_ABOUT\_ORDER; |