Kitap Market

DATABASE System

midterm project

Sanzhar Bakirbayev, Akarys Yerdali,
Mukhtar Rabayev, Yeldos Sanabek

List of Contents

Kitap Market

For bookworms only

Brand colors

#ecd7a4
#9f763a
#aa8349
#b59158
#c09f67

Business Process:

Our project's main goal is to implement a database for an online bookstore called 'Kitap Market'.

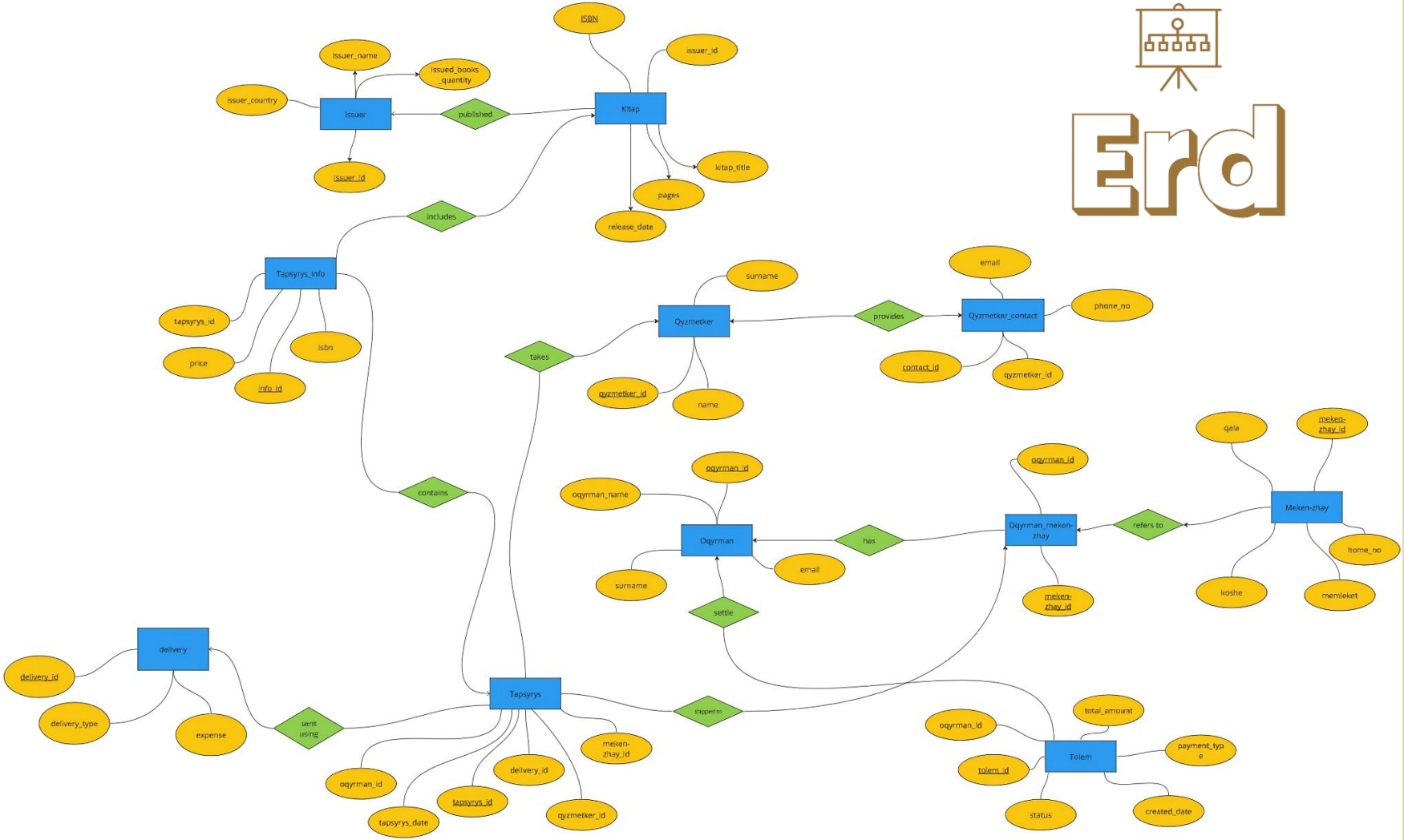
Firstly, book issuers are supplying us with books they published. After that, customers, readers could order books from our market after they chose what they liked. In order to be executed bookstore administrators must know addresses of their customers, we must mention that one reader could have more than one living address. So, the customer is making an order, and our employee takes that order, after all operations are correct, the reader makes a payment. Every employee in our company has unique contacts. After all these operations are executed we start to deliver our customers' orders with any type or transportation. And finally, we have happy readers and happy company.

EXPLANATION OF OUR ERD

We're going to design an ER or entity-link diagram, break down the different types of links, and visualize them using an example. After all, a picture is always clearer than text.

An ER diagram graphically represents the entities of a subject area, the attributes of the entities and the relationship between them

ERD attributes characterize entities, allowing users to better understand the structure of the database. The attributes contain information about the entities highlighted in the conceptual ER diagram.


Erd
ISBN
issuer_id
issuer_name
issued_books_quantity
issuer_country
Issuer
published
Kitap
kitap_title
pages
release_date
issuer_id
includes
Tapsyrys_info
tapsyrys_id
price
isbn
info_id
surname
Qyzmetker
takes
qyzmetker_id
name
email
Qyzmetker_contact
provides
phone_no
contact_id
qyzmetker_id
qala
meken-zhay_id
oqyrman_id
oqyrman_name
Oqyrman
has
email
surname
settle
contains
oqyrman_id
Oqyrman_meken-zhay
refers to
Meken-zhay
home_no
koshe
memleket
meken-zhay_id
delivery
delivery_id
delivery_type
expense
sent using
Tapsyrys
shipped to
oqyrman_id
tapsyrys_date
tapsyrys_id
delivery_id
qyzmetker_id
meken-zhay_id
total_amount
oqyrman_id
Tolem
payment_type
tolem_id
status
created_date

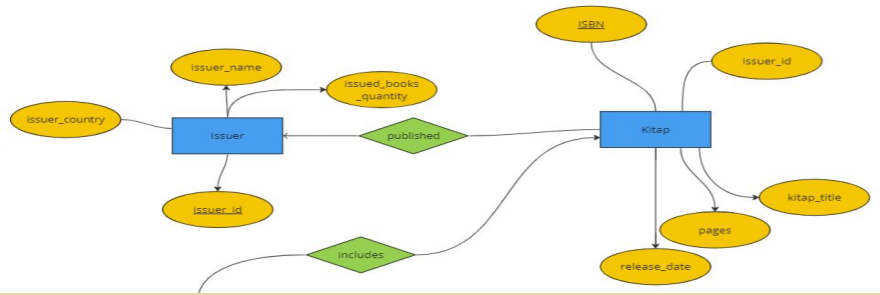
Relationships between tables

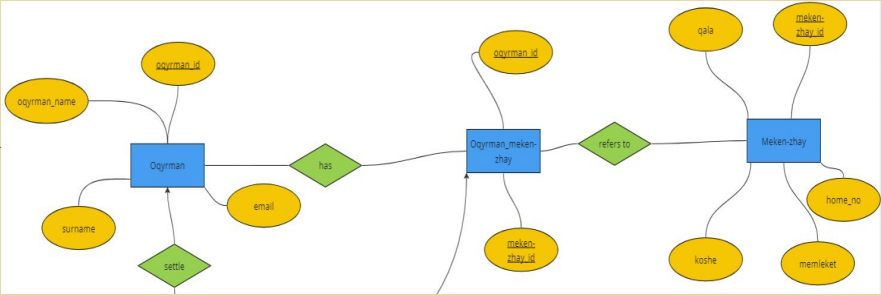
There are 10 relationships in our erd diagram. We have all types of relationships:

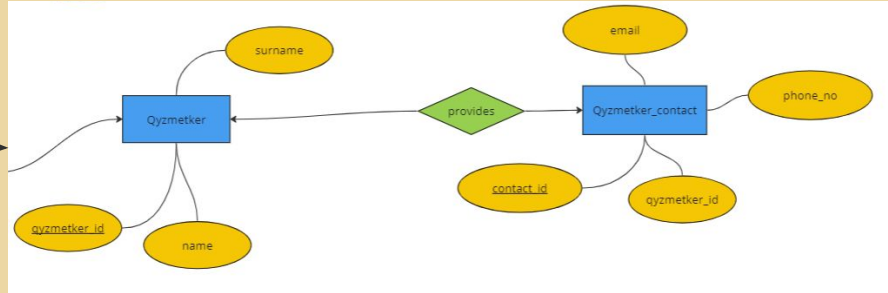
Many-to-One

One-to-one

Many-to-Many







Tables

There are 11 tables in total, here are the first 5 tables

TAPSYRYS

Columns | Data | Indexes | Constraints | Grants | Statistics | Triggers | Dependencies | DDL | Sample Queries

+ Insert Row | Columns... | Filter... | Count Rows | Load Data | Download | Refresh

OQYRMAN_ID	TAPSYRYS_DATE	TAPSYRYS_ID	DELIVERY_ID	QYZMETKER_ID	MEKEN_ZHAY_ID
1	03/28/2023	1	41	20	1
2	09/10/2022	2	29	36	2
3	11/17/2022	3	28	4	3
4	06/27/2022	4	49	25	4
5	09/13/2022	5	25	33	5
6	05/31/2022	6	11	30	6
7	03/04/2023	7	23	24	7
8	09/08/2022	8	15	5	8

QYZMETKER

Columns | Data | Indexes | Constraints | Grants | Statistics | Triggers | Dependencies | DDL | Sample Queries

+ Insert Row | Columns... | Filter... | Count Rows | Load Data | Download | Refresh

QYZMETKER_ID	NAME	SURNAME
1	Reinhard	Ormond
2	Caspar	Dybell
3	Benedetto	MacGorley
4	Sheffield	Cronshaw
5	Chrysa	Piscopo
6	Nertie	Rollinson
7	Maxwell	Botwright
8	Emmit	Limming

TOLEM

Columns | Data | Indexes | Constraints | Grants | Statistics | Triggers | Dependencies | DDL | Sample Queries

+ Insert Row | Columns... | Filter... | Count Rows | Load Data | Download | Refresh

OQYRMAN_ID	TOLEM_ID	STATUS	TOTAL_AMOUNT	PAYMENT_TYPE	CREDIT_DATE
1	1	not paid	34550	webmoney	09/04/2022
2	2	not paid	72706	qiwi wallet	08/04/2022
3	3	paid	80451	bitcoin	12/16/2022
4	4	paid	92488	credit card	04/30/2022
5	5	paid	60540	yandex wallet	03/05/2023
6	6	paid	24513	webmoney	09/14/2022
7	7	paid	74582	webmoney	11/18/2022
8	8	paid	5052	webmoney	04/09/2023

QYZMETKER_CONTACT

Columns | Data | Indexes | Constraints | Grants | Statistics | Triggers | Dependencies | DDL | Sample Queries

+ Insert Row | Columns... | Filter... | Count Rows | Load Data | Download | Refresh

CONTACT_ID	QYZMETKER_ID	EMAIL	PHONE_NO
1	25	oschapiro0@archive.org	443-921-4651
2	33	ogajownik1@slashdot.org	107-385-5048
3	21	kchelarn2@com.com	560-946-9313
4	47	hspadotto3@ucla.edu	699-491-7632
5	42	slinder4@yolasite.com	261-151-7444
6	8	tbrevetor5@scribd.com	818-993-6631
7	35	stillerton6@list-manage.com	680-342-4449
8	12	emoules7@hexun.com	624-268-7848

TAPSYRYS_INFO

Columns | Data | Indexes | Constraints | Grants | Statistics | Triggers | Dependencies | DDL | Sample Queries

+ Insert Row | Columns... | Filter... | Count Rows | Load Data | Download | Refresh

TAPSYRYS_ID	PRICE	INFO_ID	ISBN
45	1422	1	097317074-3
12	2204	2	334404136-3
24	446	3	555573638-X
34	2500	4	059963415-4
42	1918	5	696841512-7
39	380	6	662478100-9
19	732	7	644399511-2
11	1755	8	703984631-4

OQYRMAN

Columns | **Data** | Indexes | Constraints | Grants | Statistics | Triggers | Dependencies | DDL | Sample Queries

+ Insert Row | Columns... | Filter... | Count Rows | Load Data | Download | Refresh

OQYRMAN_ID	OQYRMAN_NAME	SURNAME	EMAIL
1	Cybil	Hastings	chastings0@ifeng.com
2	Dahlia	Cicculini	dcicculini1@soundcloud.com
3	Rosene	Gyse	rgyse2@symantec.com
4	Swen	Vowdon	svowdon3@blinklist.com
5	Shurlocke	Pagett	spagett4@homestead.com
6	Charin	McKee	cmckee5@foxnews.com
7	Jakob	Cuell	jcuell6@icio.us
8	Mark	Andriulis	mandriulis7@imageshack.us

MEKEN_ZHAY

Columns | **Data** | Indexes | Constraints | Grants | Statistics | Triggers | Dependencies | DDL | Sample Queries

+ Insert Row | Columns... | Filter... | Count Rows | Load Data | Download | Refresh

MEKEN_ZHAY_ID	QALA	KOSHE	HOME_NO	MEMLEKET
1	Manat	Fairview	67574	Philippines
2	Nikel'	Calypso	165	Russia
3	Shuihu	Bunker Hill	24	China
4	Santa Maria da Feira	Dryden	69	Portugal
5	Dzików Stary	Summer Ridge	51	Poland
6	Balangiga	Cody	25	Philippines
7	Anton	Hovde	0	Bulgaria
8	Dakhla	Roxbury	60244	Western Sahara

KITAP

Columns | **Data** | Indexes | Constraints | Grants | Statistics | Triggers | Dependencies | DDL | Sample Queries

+ Insert Row | Columns... | Filter... | Count Rows | Load Data | Download | Refresh

ISBN	ISSUER_ID	KITAP_TITLE	PAGES	RELEASE_DATE
097317074-3	5	Span	883	10/02/2020
334404136-3	44	Voyatouch	415	04/14/2021
555573638-X	49	Prodder	943	07/26/2020
059963415-4	37	Fintone	539	06/13/2020
696841512-7	13	Rank	1189	11/29/2020
662478100-9	45	Stronghold	1168	02/21/2021
644399511-2	5	Voyatouch	453	07/26/2020
703984631-4	18	Flexidy	641	03/31/2021

OQYRMAN_MEKEN_ZHAY

Columns | **Data** | Indexes | Constraints | Grants | Statistics | Triggers | Dependencies | DDL | Sample Queries

+ Insert Row | Columns... | Filter... | Count Rows | Load Data | Download | Refresh

MEKEN_ZHAY_ID	OQYRMAN_ID
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8

DELIVERY

Columns | **Data** | Indexes | Constraints | Grants | Statistics | Triggers | Dependencies | DDL | Sample Queries

+ Insert Row | Columns... | Filter... | Count Rows | Load Data | Download | Refresh

DELIVERY_ID	DELIVERY_TYPE	EXPENSE
1	3	669
2	2	720
3	2	201
4	2	488
5	3	667
6	2	942
7	3	651
8	3	964

ISSUER

Columns | **Data** | Indexes | Constraints | Grants | Statistics | Triggers | Dependencies | DDL | Sample Queries

+ Insert Row | Columns... | Filter... | Count Rows | Load Data | Download | Refresh

ISSUER_ID	ISSUER_COUNTRY	ISSUER_NAME	ISSUED_BOOKS_QUANTITY
1	Greece	Kozey, Durgan and Walter	3
2	Greece	Beahan-Buckridge	6
3	Afghanistan	Purdy LLC	5
4	Mexico	Skiles, Gorczany and Weber	9
5	Canada	Rau LLC	10
6	Indonesia	OConner-Graham	2
7	Indonesia	Ritchie LLC	1
8	Czech Republic	Kovacek LLC	2

table standard

All the tables were created with the corresponding ERD diagram. And it has a logical relationship with each other. Each table has a primary key and some have a Unique case. Tables and data that are stored inside are a very important part of the DataBase. Because a table without data doesn't make sense. You also need to follow the rules so that everything works as it should.

Tables. Description

With the help of SQL queries, we created all these 11 tables. Each table contains 50 rows of data. Let's look at one of the SQL queries for creating a table.

```
CREATE TABLE qyzmetker_contact
( contact_id number(10) NOT NULL PRIMARY KEY,
  qyzmetker_id number(10) NOT NULL,
  email VARCHAR2(50) NOT NULL,
  phone_no VARCHAR2(50) NOT NULL,
  FOREIGN KEY (qyzmetker_id) REFERENCES
qyzmetker(qyzmetker_id)
);
```

Creating the ‘qyzmetker_contact’ table with the following attributes:
Contact_id; qyzmetker_id; email; phone_no;

The column can only store numbers, up to 10 digits.

The column stores everything as a ‘string’ and up to 50 character.

They can't all be empty.

defines a foreign key for the "qyzmetker_id" column, which references the "qyzmetker_id" column of the "qyzmetker" table.

Pl/sql

Requests and functions that are available in our database

There are 5 in total

This process classifies our books by publishers so that you can view how many books from each publisher we have in our store.
In order to do this, we use an INNER JOIN to integrate the data from the KITAP and ISSUER tables.
Then we merely use GROUP BY to sort by issuers.To order publishers by the quantity of books they have published, use ORDER BY.

Here is sample query for checking this procedure:
BEGIN
kitap_by_issuer;
END;

```
create or replace PROCEDURE kitap_by_issuer IS
BEGIN
  FOR k IN (SELECT i.issuer_name, COUNT(b.isbn) AS total_books
            FROM kitap b
            INNER JOIN issuer i ON b.issuer_id = i.issuer_id
            GROUP BY i.issuer_name
            ORDER BY total_books)
  LOOP
    DBMS_OUTPUT.PUT_LINE(k.issuer_name || ' issued ' || k.total_books || ' kitap.');
  END LOOP;
END;
```

```
BEGIN
kitap_by_issuer;
END;
```

Results | Explain | Describe | Saved SQL | History

Describe

```
Will Group issued 1 kitap.
Schaden-Lowe issued 1 kitap.
Nitzsche-Waters issued 1 kitap.
Cummings, Hoppe and Stroman issued 1 kitap.
Hamill, Boyle and Kshlerin issued 1 kitap.
Kilback, Crist and Bergnaum issued 1 kitap.
Emard, Paucek and Cummerata issued 1 kitap.
Kozey, Durgan and Walter issued 1 kitap.
Toy and Sons issued 1 kitap.
McGlynn-McCullough issued 1 kitap.
Koepp and Sons issued 1 kitap.
Schiller, Fadel and Marks issued 1 kitap.
Orn and Sons issued 1 kitap.
Champlin-Ullrich issued 1 kitap.
Deckow, Batz and Goodwin issued 1 kitap.
Ritchie, Heidenreich and Smith issued 1 kitap.
Walsh, Marquardt and Morar issued 1 kitap.
Hills Group issued 1 kitap.
Bednar, Rippin and Streich issued 1 kitap.
```

```
create or replace FUNCTION count_oqyrmans
RETURN NUMBER
IS
  t_count NUMBER;
BEGIN
  SELECT COUNT(*) INTO t_count FROM oqyrman;
  RETURN t_count;
END;
```

This function totals the number of active clients that are listed in our database.
We must establish a variable (in this case, t_count) number in order to save the total number of purchasers.The number of rows (or purchasers, depending on your perspective) is then saved in our newly generated variable using COUNT(*). Then we give it back. That is how our role operates.

```
DECLARE
   o_counts number(2);
BEGIN
   o_counts := count_oqyrmans();
   dbms_output.put_line('Total number of Oqyrmans: ' || o_counts);
END;
```

Results | Explain | Describe | Saved SQL | History

```
Total number of Oqyrmans: 20

Statement processed.
```

```oracle
create or replace PROCEDURE issuer_updating (
  c_issuer_id IN Issuer.issuer_id%TYPE,
  c_issued_books_quantity IN Issuer.issued_books_quantity%TYPE

)
IS
BEGIN
  UPDATE Issuer
  SET issued_books_quantity = c_issued_books_quantity
  WHERE issuer_id = c_issuer_id;
  DBMS_OUTPUT.PUT_LINE('Number of rows affected: ' || SQL%ROWCOUNT);
END;
```

```oracle
DECLARE
  c_issuer_id Issuer.issuer_id%TYPE := 1;
  c_issued_books_quantity Issuer.issued_books_quantity%TYPE := 5;
BEGIN
  issuer_updating(c_issuer_id, c_issued_books_quantity);
END;
```

Results | Explain | Describe | Saved SQL | History

```
Number of rows affected: 1

Statement processed.
```

The number of updated publisher data is counted in this method. The technique counts the number of modified rows whenever a publisher makes a new book available for purchase or withdraws an existing title from sale.

To accomplish this, we require the incredibly useful system function SQL%ROWCOUNT, which returns the number of rows that the query has touched.

```oracle
DECLARE
   c_koshe Meken_zhay.koshe%type;
   c_home_no Meken_zhay.home_no%type;
   c_qala Meken_zhay.qala%type;
   c_memleket Meken_Zhay.memleket%type;
   ex_short_street_name  EXCEPTION;
BEGIN
   FOR i IN (SELECT meken_zhay_id, koshe, home_no, qala, memleket FROM Meken_zhay ORDER BY meken_zhay_id)
   LOOP
      c_koshe := i.koshe;
      c_home_no := i.home_no;
      c_qala := i.qala;
      c_memleket := i.memleket;

      IF LENGTH(c_koshe) < 3 THEN
         RAISE ex_short_street_name;
      ELSE
         DBMS_OUTPUT.PUT_LINE ('Street: ' || c_koshe);
         DBMS_OUTPUT.PUT_LINE ('Home number: ' || c_home_no);
         DBMS_OUTPUT.PUT_LINE ('City: ' || c_qala);
         DBMS_OUTPUT.PUT_LINE ('Country: ' || c_memleket);
      END IF;
   END LOOP;
EXCEPTION
   WHEN ex_short_street_name THEN
      dbms_output.put_line('Too short street name');
END;
```

When you perform a SELECT query, the system raises an exception and displays a notice if it discovers a street name that is less than 6 characters long. We use a FOR loop to process our data in order to make the exception function. The query examines every line. If everything is in order, the query displays the user's complete address. An exception is thrown if the street has fewer than 6 characters.

```oracle
DECLARE
   c_koshe Meken_zhay.koshe%type;
   c_home_no Meken_zhay.home_no%type;
   c_qala Meken_zhay.qala%type;
   c_memleket Meken_Zhay.memleket%type;
   ex_short_street_name  EXCEPTION;
BEGIN
   FOR i IN (SELECT meken_zhay_id, koshe, home_no, qala, memleket FROM Meken_zhay ORDER BY meken_zhay_id)
   LOOP
      c_koshe := i.koshe;
      c_home_no := i.home_no;
```

Results | Explain | Describe | Saved SQL | History

```
City: Shuihu
Country: China
Street: Dryden
Home number: 69
City: Santa Maria da Feira
Country: Portugal
Street: Summer Ridge
Home number: 51
City: Dzików Stary
Country: Poland
Too short street name
```

```oracle
create or replace TRIGGER tolem_trig
BEFORE INSERT ON Tolem
FOR EACH ROW
DECLARE
ccount number;
PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
 SELECT COUNT(*) into ccount
 FROM TOLEM;
 commit;
 DBMS_OUTPUT.PUT_LINE('Number of rows in Payment table: ' || ccount);
END;
```

Before a new row is added to the TOLEM table, this trigger is fired. i.e., the payment information is written to the table when the customer makes a payment. Additionally, a trigger is activated to count all payments prior to

```oracle
INSERt INTO Tolem(oqyrman_id, tolem_id, status, total_amount, payment_type, credit_date)
            VALUES(17, 51, 'paid', 2000, 'credit card', '09/04/2022');
```

Results | Explain | Describe | Saved SQL | History

```
Number of rows in Payment table: 50

1 row(s) inserted.
```

CONCLUSION

We created this database for Kitap Market bookstore. It will be used for both administrators and workers. We believe that this database is optimized and user-friendly for this database.

Thank you for your attention. This concludes our presentation.