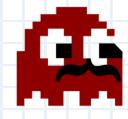




DATABASE

از زمین خاکی



فهرست مطالب

□ تاریخچه

□ مفاهیم

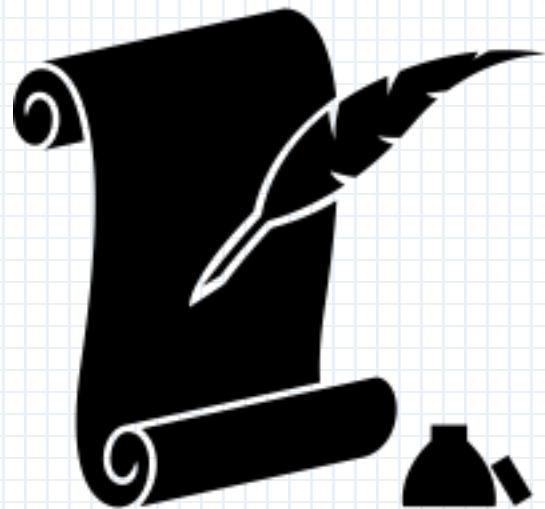
□ طراحی

DBMS □

SQL □



تاریخچه پاگاه داده



HISTORY



ذخیره سازی



HISTORY



اشتران گذاری

HISTORY



بازیابی

HISTORY



نام	نوع محصول	ادرس	مقدار
كمبوجيه اول	گندم	شوش جنب اکبرجوچه	۱۰۰ کيسه
سورنا سوم	جو دوسر مرغوب	نقش رستم پلاک ۲	۴۰ کيسه

سيستم اطلاعاتي انبارداري همايوني



HISTORY



اختراع كاغذ



HISTORY



ثبتنام دانشآموزان در مدرسه

HISTORY



معایب این روش ذخیره‌سازی



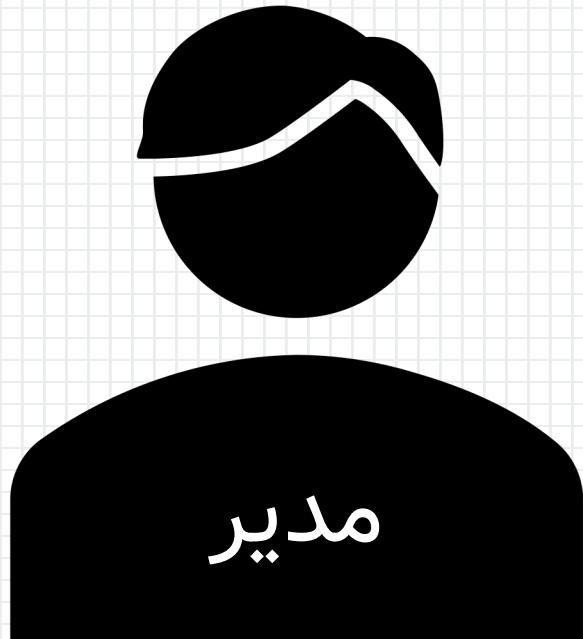
HISTORY



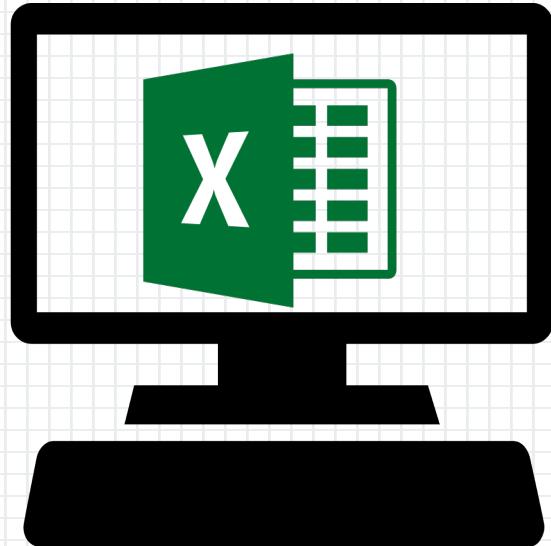
اختراع كامپيوتر



HISTORY

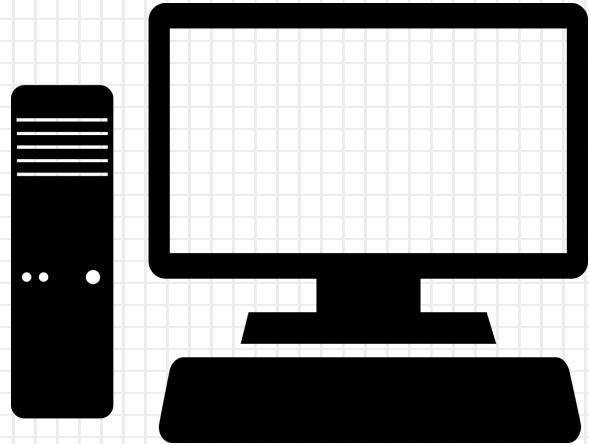


نام	ادرس	پایه	معدل
عباس عباسی	محمدآباد	دهم	۱۹
محمد محمدی	عباس آباد	یازدهم	۱۹.۹۹



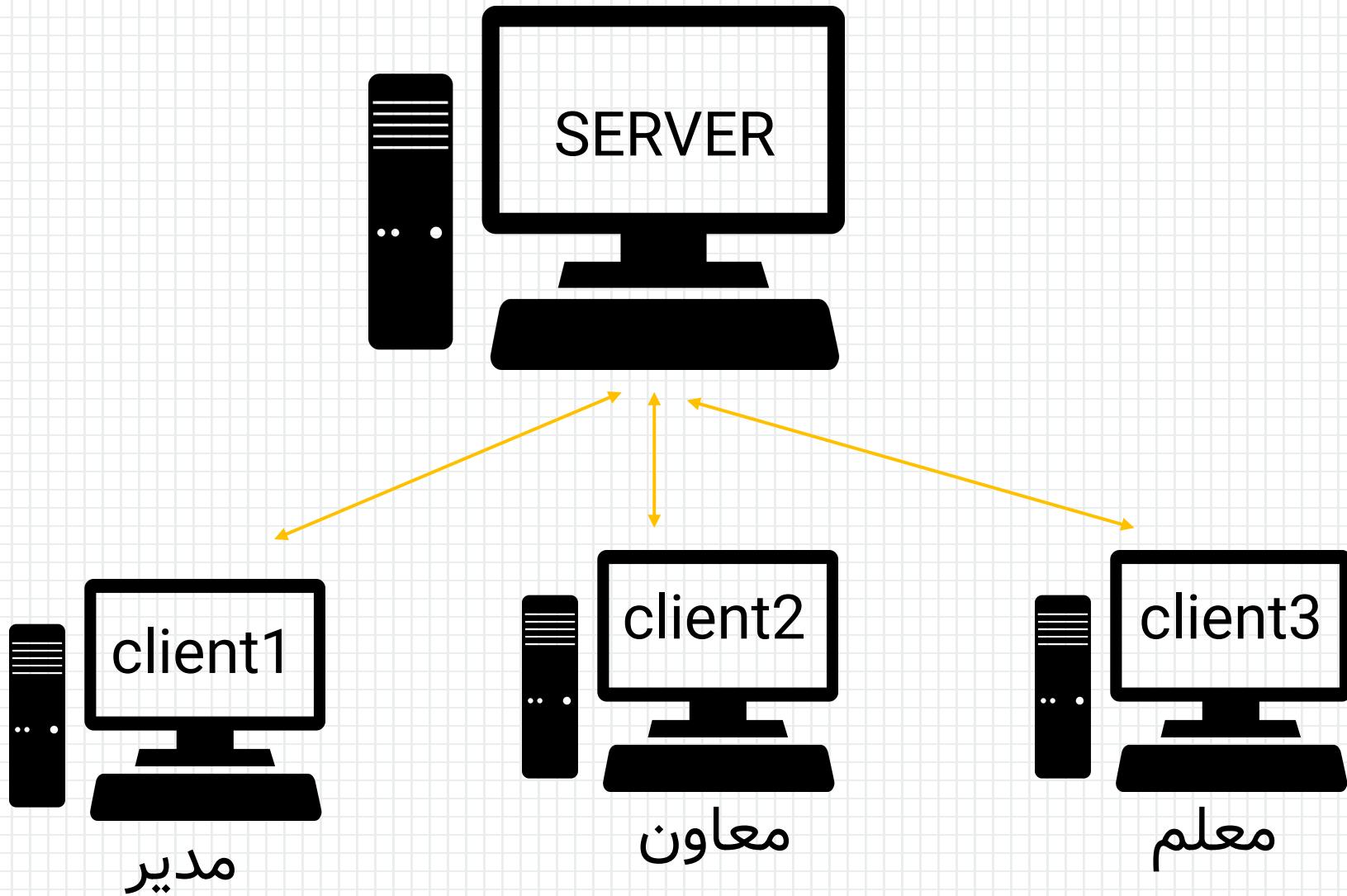
ثبتنام دانشآموزان در مدرسه



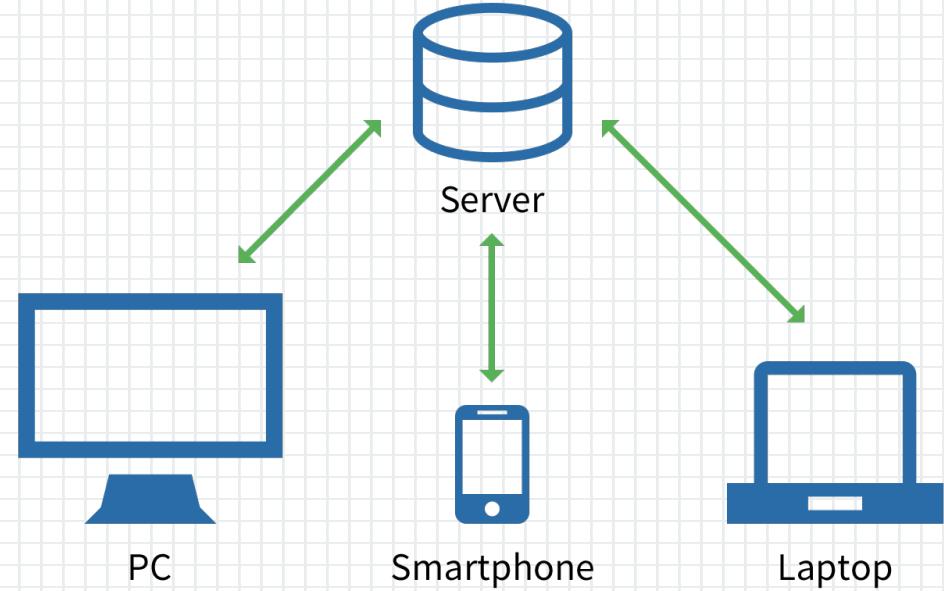


معایب این روش ذخیره‌سازی

HISTORY



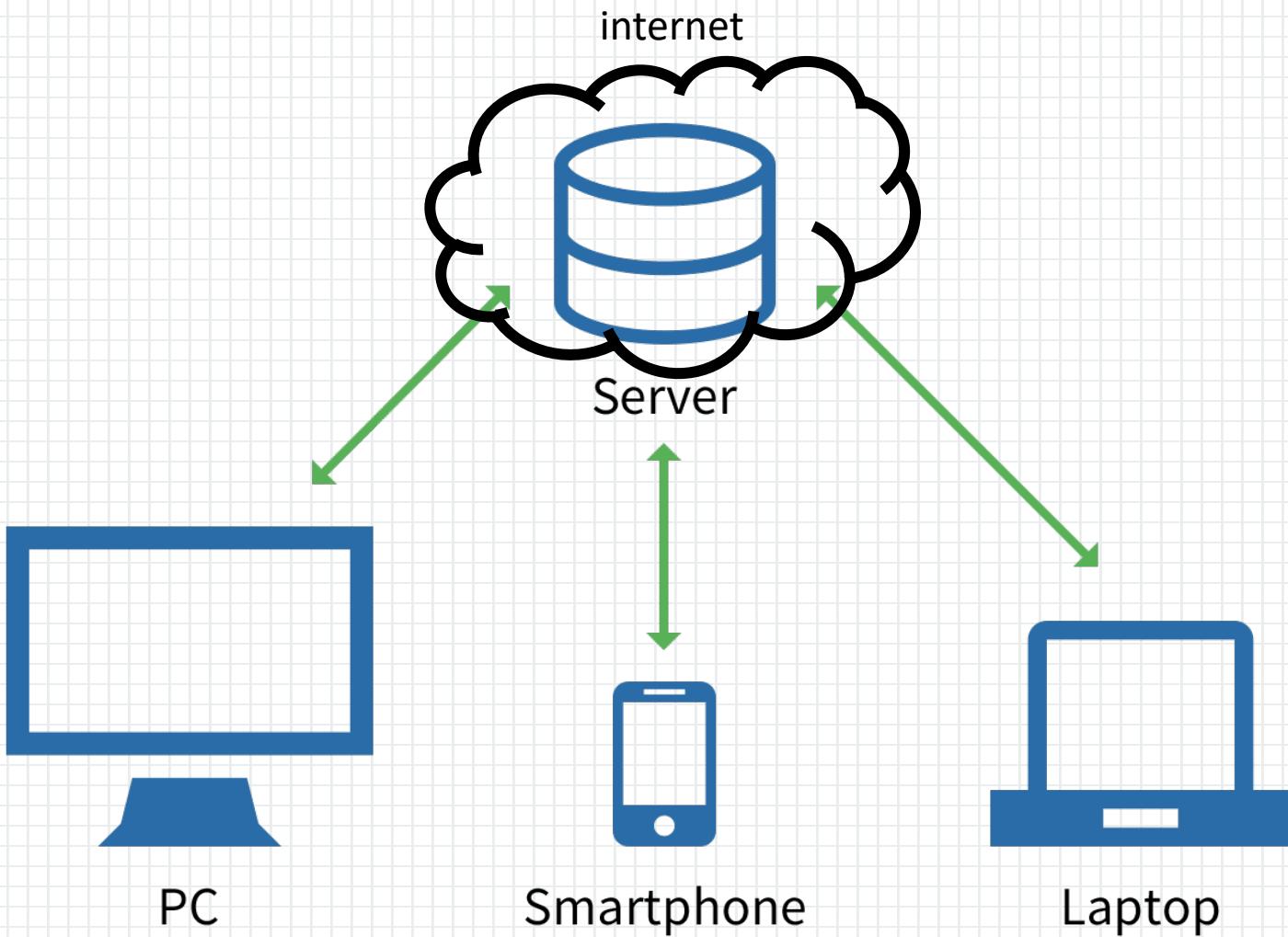
HISTORY



معایب این روش ذخیره‌سازی



HISTORY



HISTORY



HISTORY



HISTORY



HISTORY



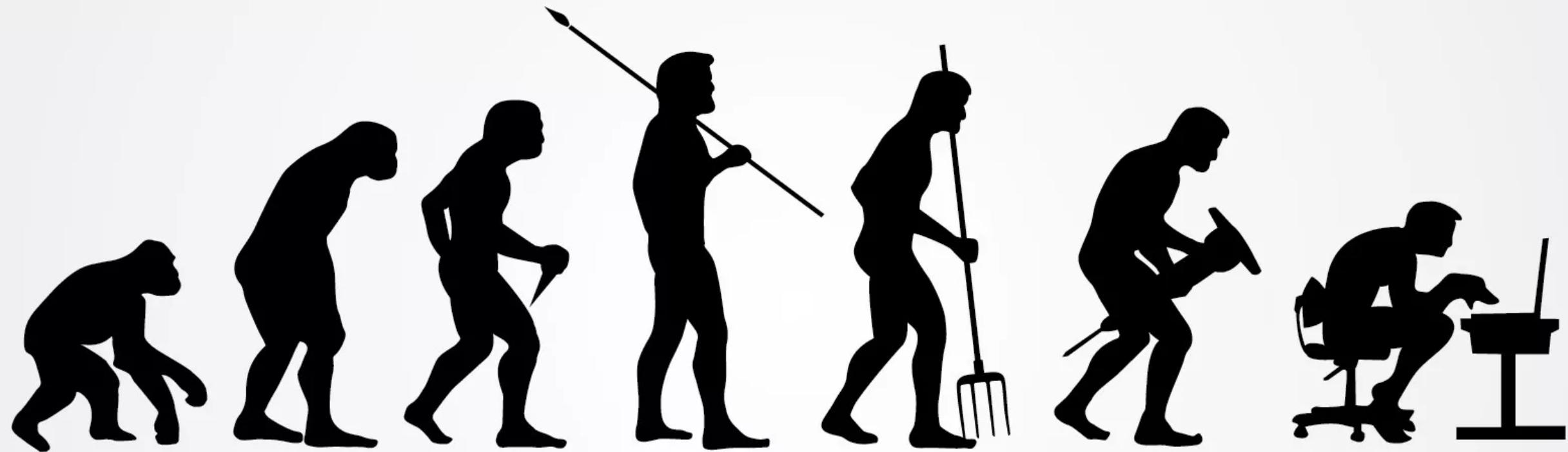
معایب این روش ذخیره‌سازی



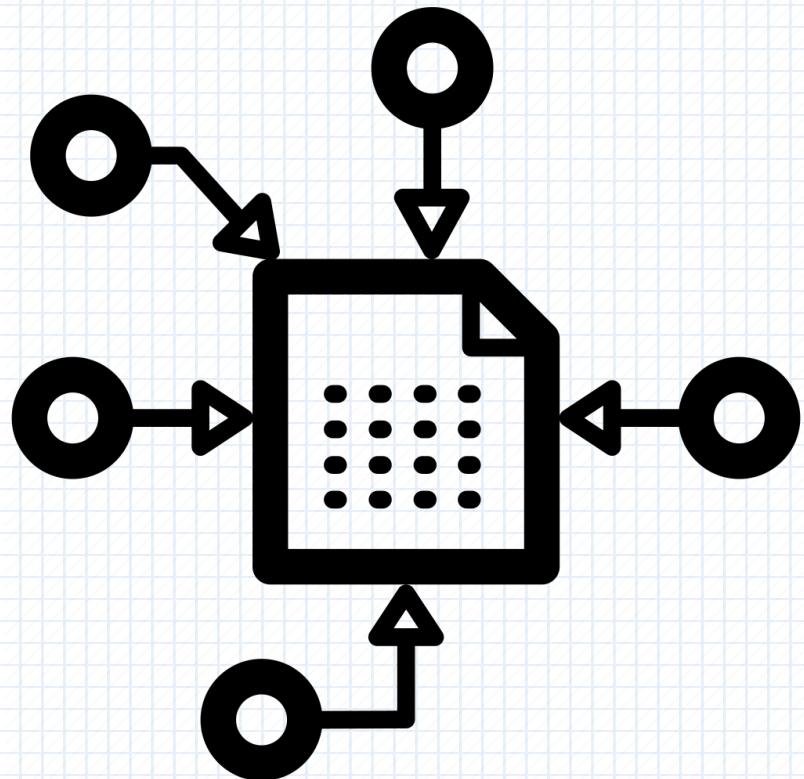
- Redundancy (داده‌های تکراری)
- Conflict (تداخل)
- Update (به‌روز رسانی)
- Security (امنیت)



HISTORY



مفاهیم پایگاه داده



داده:



همهی دانسته‌ها، آگاهی‌ها، داشته‌ها، آمارها، شناسه‌ها، پیشینه‌ها و پنداشته‌ها

پایگاه داده:



مجموعه ای منظم و سازمان یافته از داده های ذخیره شده و الکترونیکی در سیستم رایانه ای است.

محیط عملیاتی:



محلي که میخواهیم برای آن پایگاه داده ایجاد کنیم، یک محیط عملیاتی است.

Concepts

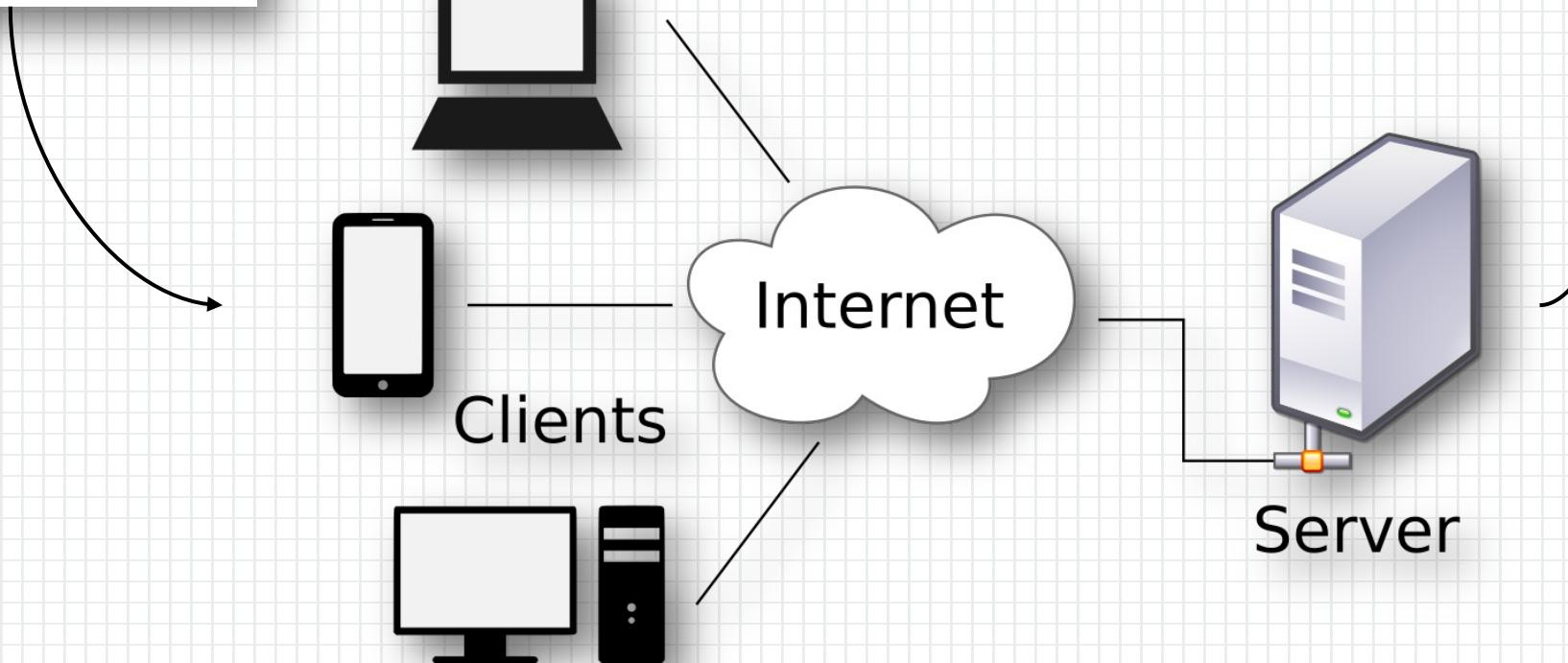
Instagram

Phone number, username or email

Password

Forgot password?

Log In



نام	نام کاربری	ایمیل	رمز
عباس عباسی	abbas	abas@gmail.com	1234
محمد محمدی	mamad	mmd@gmail.com	1111

Concepts

کاربران

نام	نام کاربری	ایمیل	رمز
عباس عباسی	abbas	۴۴۴۴
محمد محمدی	mamad

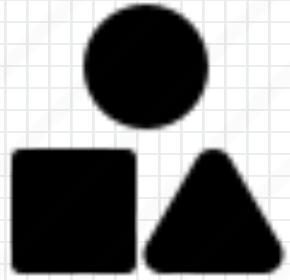
پست

امكان کامت	مکان	کپشن	عکس
خیر	برلین	یه روز خوب ...	عکس
بله	پاریس	عکس

اطلاعات کاربران

تعداد فالوینگ	تعداد فالور	بیو	عکس پروفایل
۰	۰	ارتش تک نفره	عکس
۱۰	۴۵۵	عکس

موجودیت یا :Entity



هر شخص، محل، شیء (object) یا مفهومی در دنیای واقعی که میخواهیم درباره آن اطلاعاتی را در پایگاه داده ذخیره کنیم، موجودیت نام دارد.

Concepts

Entity

کاربران

نام	نام کاربری	ایمیل	رمز
عباس عباسی	abbas	۴۴۴۴
محمد محمدی	mamad

Entity

پست

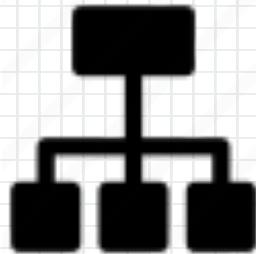
امكان کامنت	مکان	کپشن	عکس
خیر	برلین	یه روز خوب ...	عکس
بله	پاریس	عکس

Entity

اطلاعات کاربران

تعداد فالوینگ	تعداد فالور	بیو	عکس پروفایل
۰	۰	ارتش تک نفره	عکس
۱۰	۴۵۵	عکس

صفت یا :Attribute



هر موجودیت دارای تعدادی صفت است. صفت برای بیان ویژگیهای یک موجودیت استفاده می شود.

Concepts

Attributes

کاربران

نام	نام کاربری	ایمیل	رمز
عباس عباسی	abbas	۴۴۴۴
محمد محمدی	mamad

پست

امكان کامت	عکس	کپشن	مکان
خیر	عکس	یه روز خوب ...	برلین
بله	عکس	پاریس

اطلاعات کاربران

تعداد فالوینگ	تعداد فالور	بیو	عکس پروفایل
۰	۰	ارتش تک نفره	عکس
۱۰	۴۵۵	عکس

صفت کلیدی یا :primary key

صفتی که میتوان با استفاده از آن، موجودیت را به صورت یکتا و منحصر به فرد شناسایی کرد، صفت کلیدی میگویند.



Concepts

نام	نام کاربری	ایمیل	رمز
عباس عباسی	abbas	abs@gmail.com	1111
محمد محمدی	mamad	mmd@gmail.com	1234

Concepts

Primary key			
نام	نام کاربری	ایمیل	رمز
عباس عباسی	abbas	abs@gmail.com	1111
محمد محمدی	mamad	mmd@gmail.com	1234

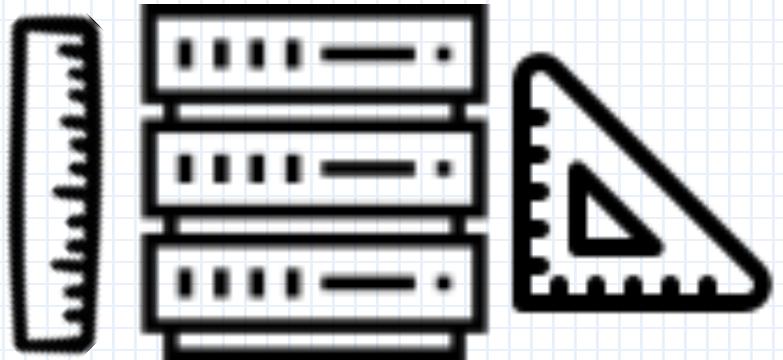


Concepts

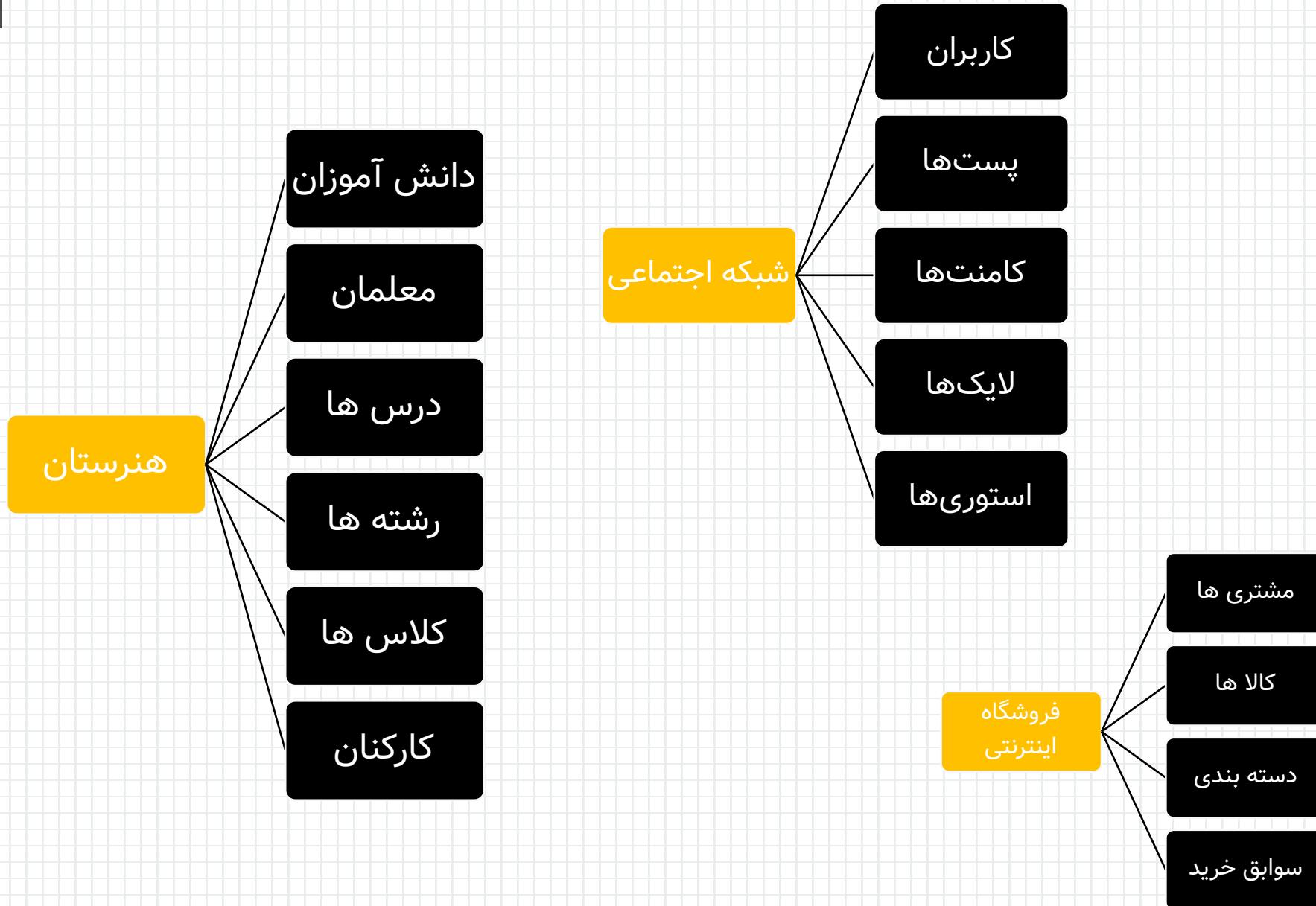
Primary key		Primary key	
نام	نام کاربری	ایمیل	رمز
عباس عباسی	abbas	abs@gmail.com	1111
محمد محمدی	mamad	mmd@gmail.com	1234



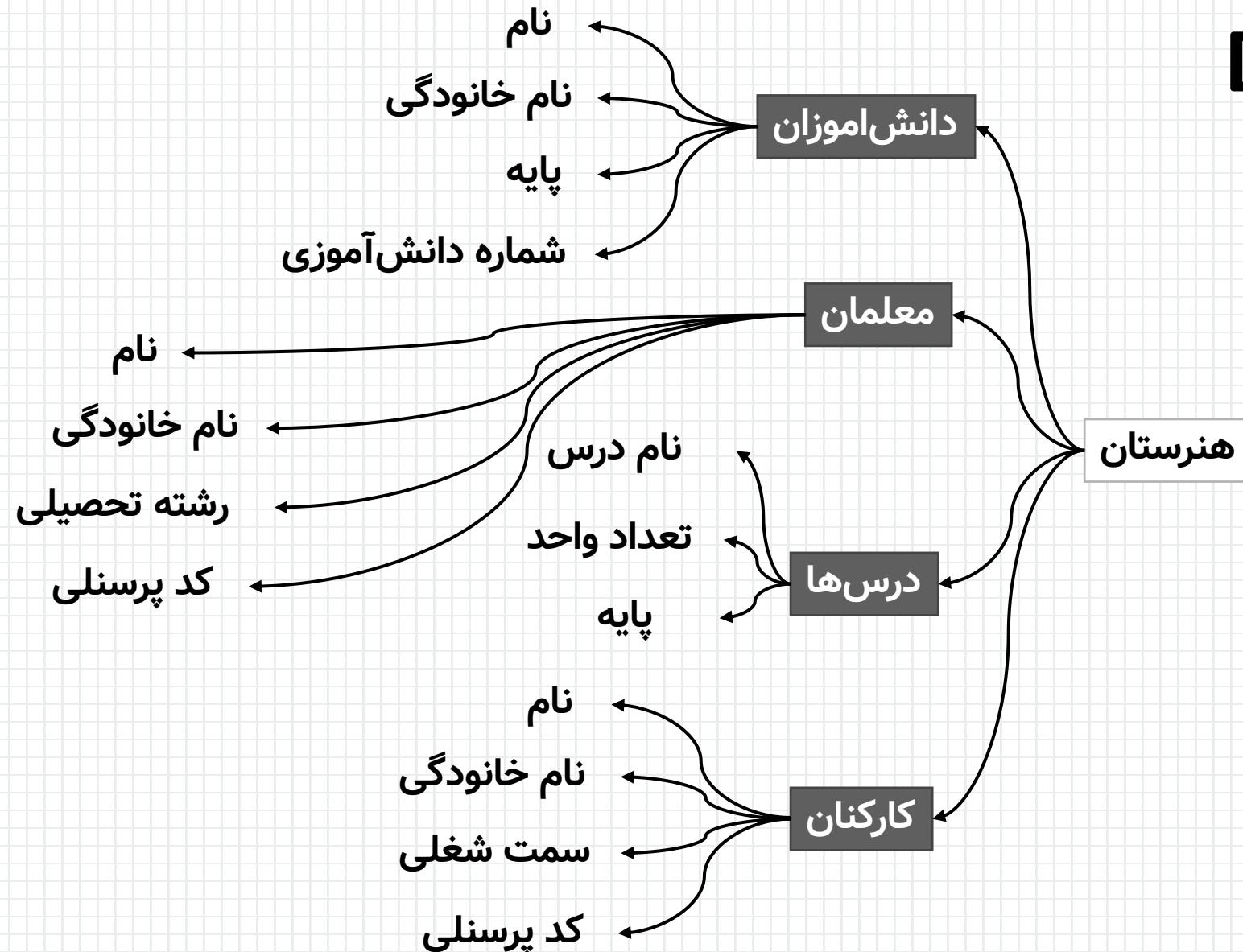
طراحی پایگاه داده



Design

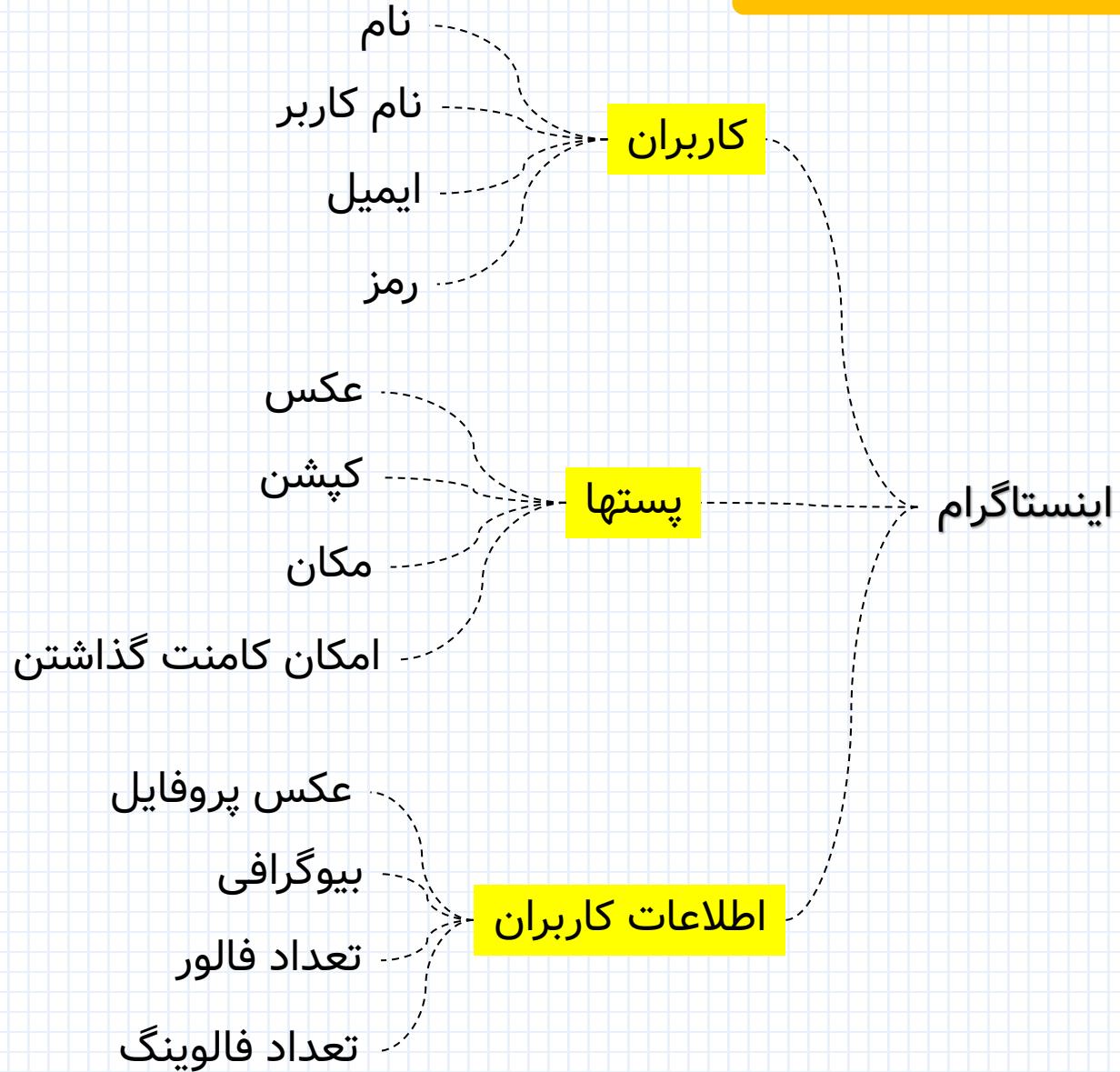
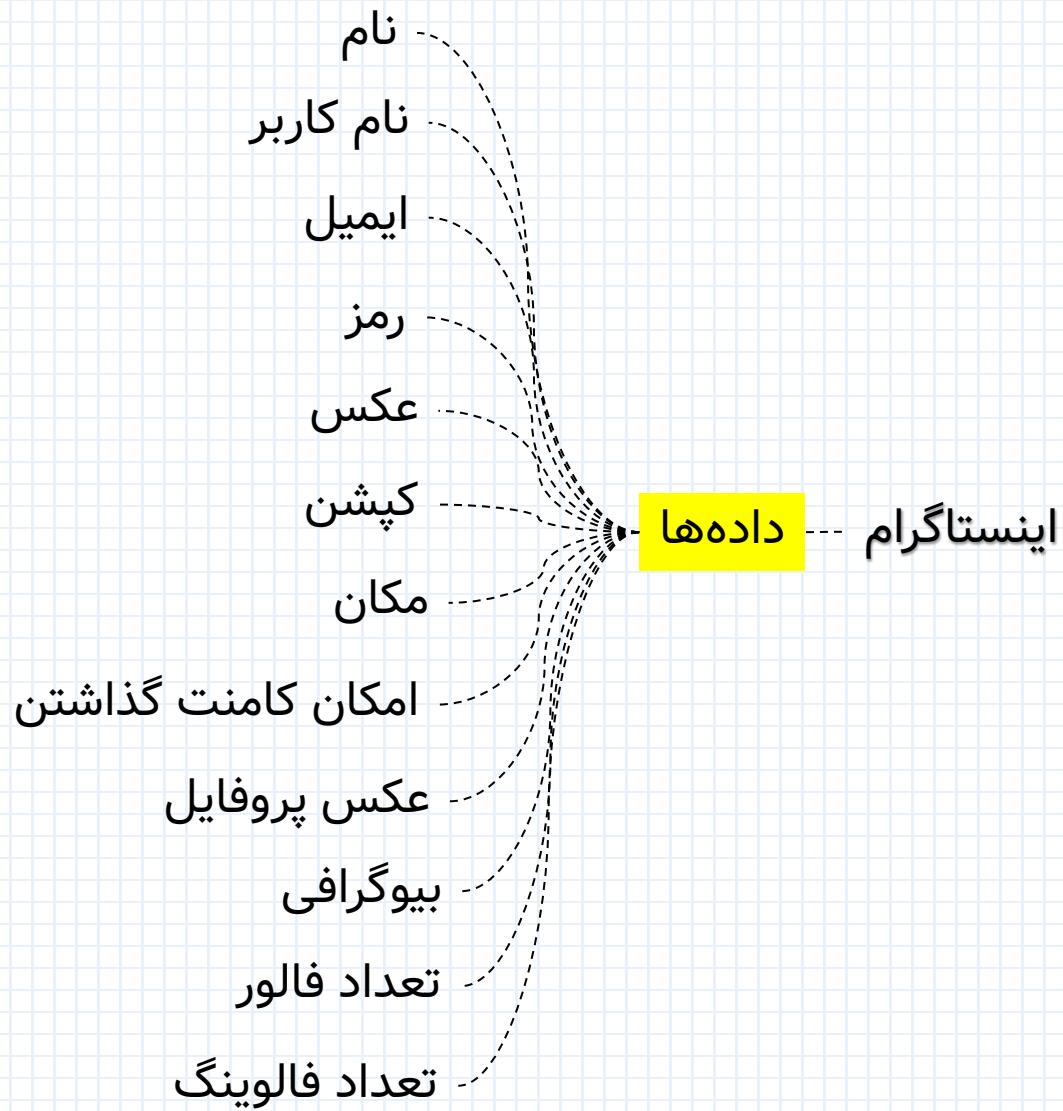


Design



تحليل دوم با ۱ موجودیت

تحلیل اول با ۳ موجودیت



اینستاگرام

داده ها

امکان کامنت	مکان	کپشن	عکس	تعداد فالوینگ	تعداد فالور	بیو	عکس پروفایل	رمز	ایمیل	نام کاربری	نام
خیر	شیروان	یه روز خوب ...	عکس ۱	۲۰	۲۰	ارتش تک نفره	عکس	۴۴۴۴	abass@gmail.com	abbas	عباس عباسی
خیر	فاروج	رفیق خوب	عکس ۲	۲۰	۲۰	ارتش تک نفره	عکس	۴۴۴۴	abass@gmail.com	abbas	عباس عباسی
بله	شیروان	تنها باش	عکس ۳	۲۰	۲۰	ارتش تک نفره	عکس	۴۴۴۴	abass@gmail.com	abbas	عباس عباسی
بله	پاریس	تصویر ۱	۱۱۱۱	mamd@gmail.com	mamad	محمد محمدی
بله	سن-ملو	تصویر ۲	۱۱۱۱	mamd@gmail.com	mamad	محمد محمدی



اینستاگرام

داده ها

امکان کامنت	مکان	کپشن	عکس	تعداد فالوینگ	تعداد فالور	بیو	عکس پروفایل	رمز	ایمیل	نام کاربری	نام
خیر	شیروان	یه روز خوب ...	عکس ۱	۲۰	۲۰	ارتش تک نفره	عکس	۴۴۴۴	abass@gmail.com	abbas	عباس عباسی
خیر	فاروج	رفیق خوب	عکس ۲	۲۰	۲۰	ارتش تک نفره	عکس	۴۴۴۴	abass@gmail.com	abbas	عباس عباسی
بله	شیروان	تنها باش	عکس ۳	۲۰	۲۰	ارتش تک نفره	عکس	۴۴۴۴	abass@gmail.com	abbas	عباس عباسی
بله	پاریس	تصویر ۱	۱۱۱۱	mamad@gmail.com	mamad	محمد محمدی
بله	سن-ملو	تصویر ۲	۱۱۱۱	mamad@gmail.com	mamad	محمد محمدی

۶۰



اینستاگرام

کاربران

نام	نام کاربری	ایمیل	رمز
عباس عباسی	abbas	abass@gmail.com	۴۴۴۴
محمد محمدی	mamad	mamd@gmail.com	۱۱۱۱

اطلاعات کاربران

عکس پروفایل	بیو	ارتش تک نفره	تعداد فالور	تعداد فالوینگ
عکس			۲۰	۲۰

پست

امكان کامنت	مکان	کپشن	عکس
خیر	شیروان	یه روز خوب ...	عکس ۱
خیر	فاروج	رفیق خوب	عکس ۲
بله	شیروان	تنها باش	عکس ۳
بله	پاریس	تصویر ۱
بله	سن-ملو	تصویر ۲



اینستاگرام

کاربران

نام	نام کاربری	ایمیل	رمز
عباس عباسی	abbas	abass@gmail.com	۴۴۴۴
محمد محمدی	mamad	mamd@gmail.com	۱۱۱۱

اطلاعات کاربران

عکس پروفایل	بیو	ارتش تک نفره	تعداد فالور	تعداد فالوینگ
عکس			۲۰	۲۰

پست

امكان کامنت	مکان	کپشن	عکس
خیر	شیروان	یه روز خوب ...	عکس ۱
خیر	فاروج	رفیق خوب	عکس ۲
بله	شیروان	تنها باش	عکس ۳
بله	پاریس	تصویر ۱
بله	سن-ملو	تصویر ۲

32



گذاشتن

کاربران

نام	نام کاربری	ایمیل	رمز
عباس عباسی	abbas	abass@gmail	۴۴۴۴
محمد محمدی	mamad	mamd@gmail	۱۱۱۱

پست

امكان کامنت	عکس	کپشن	مکان
خیر	عکس ۱	یه روز خوب	شیروان
خیر	عکس ۲	رفیق خوب	فاروج
بله	عکس ۳	تنها باش	شیروان
بله	تصویر ۱	پاریس
بله	تصویر ۲	سن-ملو

لایک کردن

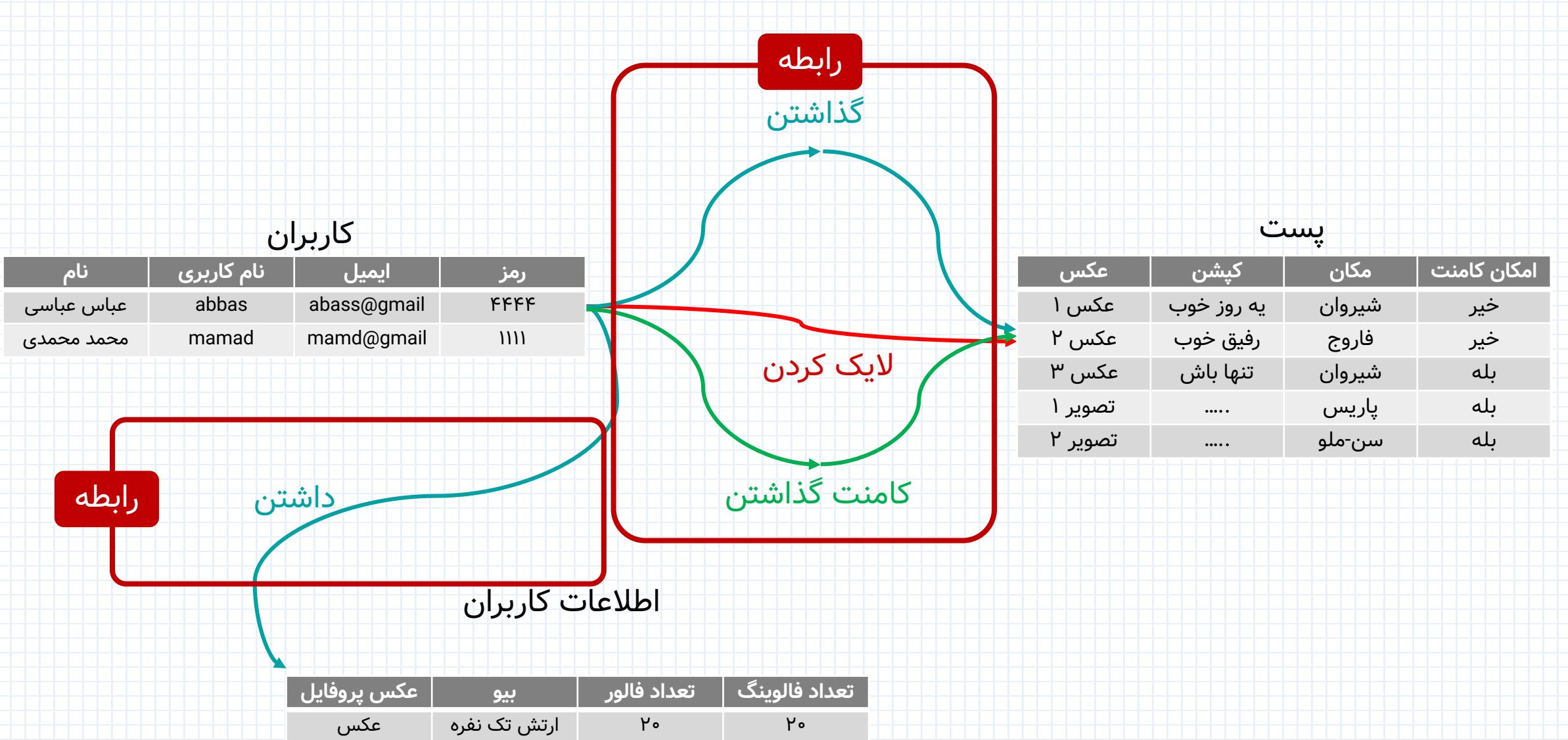
کامنت گذاشتن

داشتن

اطلاعات کاربران

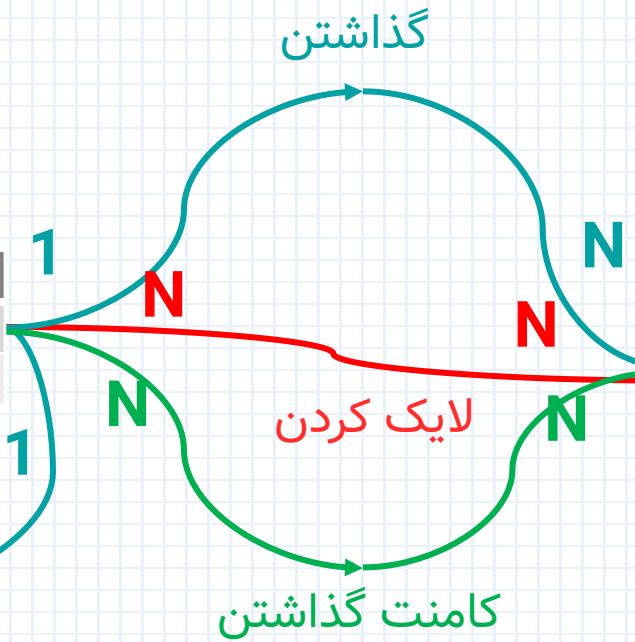
تعداد فالوینگ	تعداد فالور	بیو	عکس پروفایل
۲۰	۲۰	ارتش تک نفره	عکس





انواع روابط

کاربران			
نام	نام کاربری	ایمیل	رمز
عباس عباسی	abbas	abass@gmail	۴۴۴۴
محمد محمدی	mamad	mamd@gmail	۱۱۱۱



اطلاعات کاربران

تعداد فالوینگ	تعداد فالور	بیو	عکس پروفایل
۲۰	۲۰	ارتش تک نفره	عکس

امكان کامنت	پست	مکان	کیشن	عکس
خیر	شیروان	یه روز خوب	عکس ۱
خیر	فاروج	رفیق خوب	عکس ۲
بله	شیروان	تنها باش	عکس ۳
بله	پاریس	تصویر ۱
بله	سن-ملو	تصویر ۲

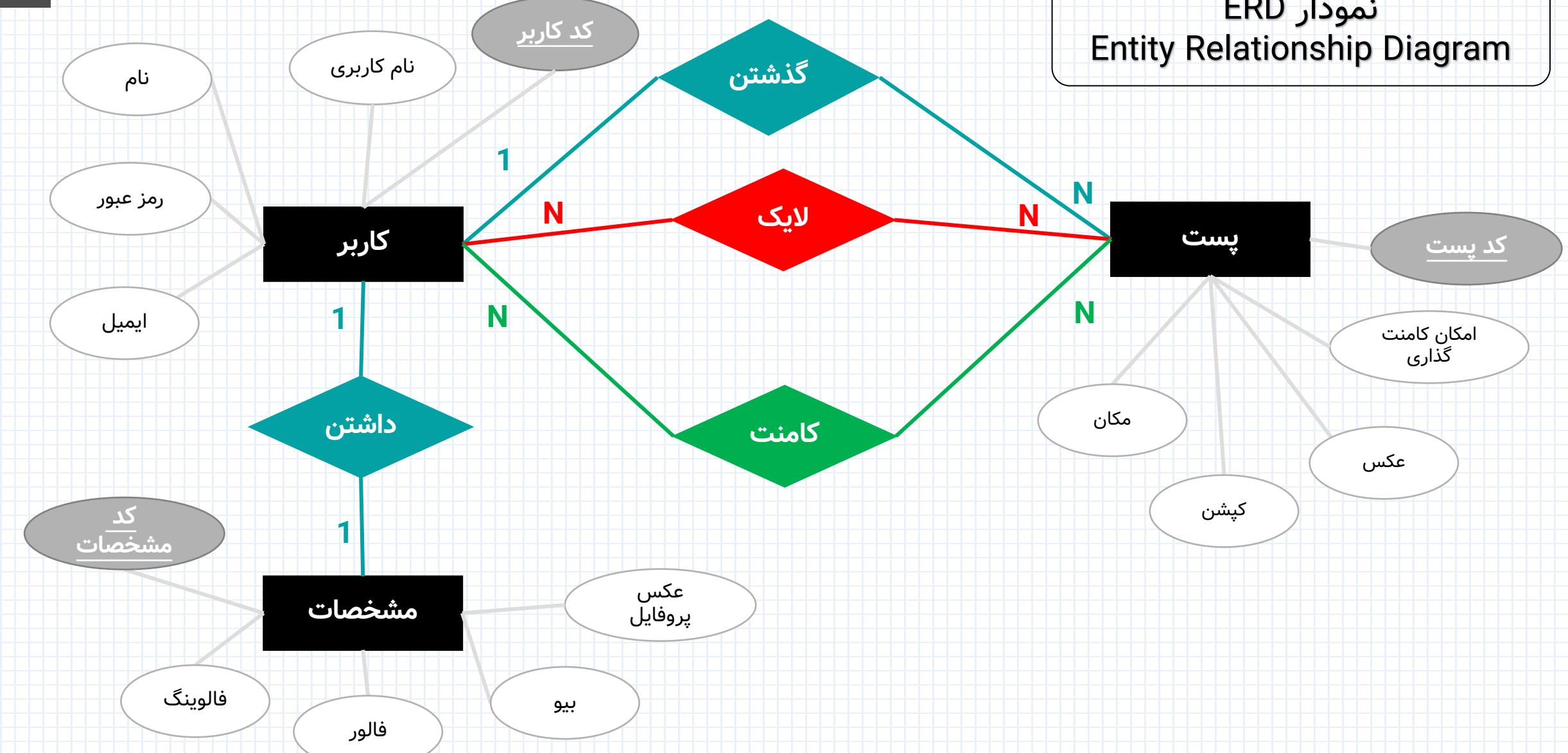


نکات طراحی

- از بین بردن داده های تکراری
- دسترسی ساده تر به اطلاعات
- تعیین هدف پایگاه داده برای اپلیکیشن خاص و نیازمندی
- مشخص کردن کلیداصلی هر جدول
- نحوه ارتباط هر جدول با جدول دیگر

نمودار ERD

Entity Relationship Diagram

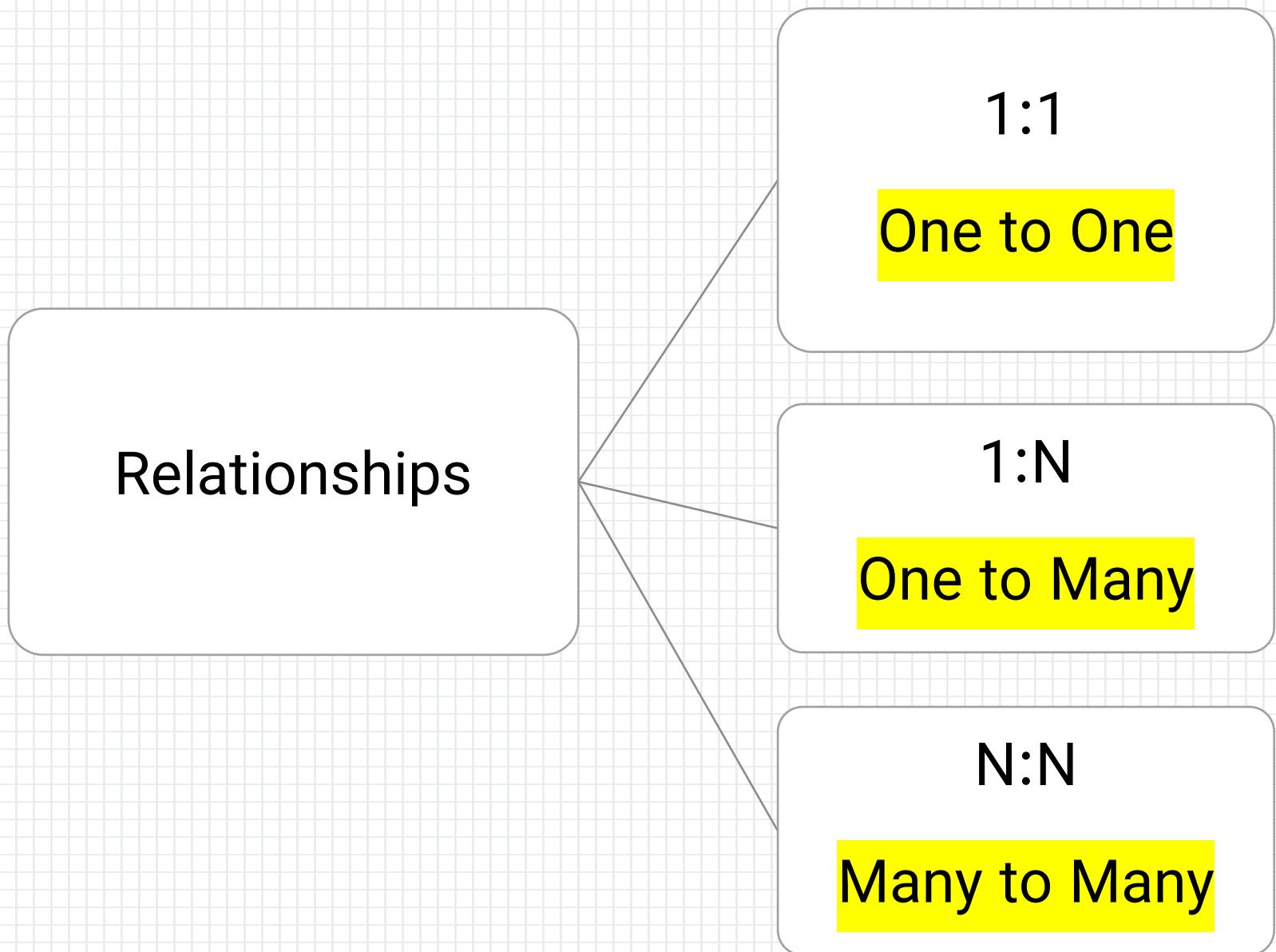


Design

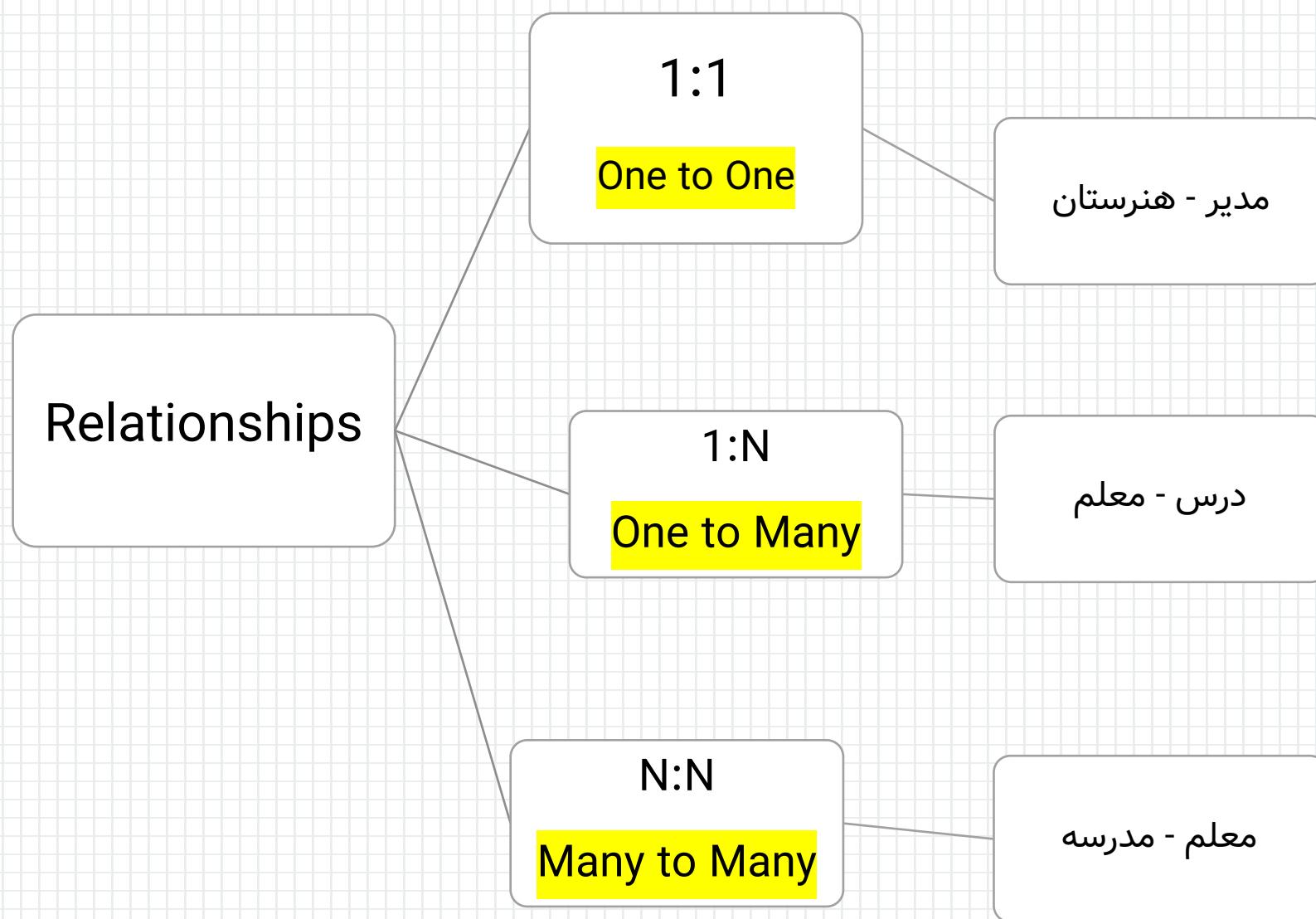
طراحی نمودار ERD هنرستان



Design



Design



کلید خارجی یا :Foreign key



کلید خارجی به فیلدی از یک جدول گفته میشود که رکوردي را به صورت منحصر به فرد و یکتا در جدول دیگر مشخص میکند. به عبارت دیگر، کلید خارجی در جدول دوم تعریف میشود اما به کلید اصلی در جدول اول اشاره دارد و ارتباط بین جدولها را امکانپذیر میسازد.

M:N

Many to Many

- **Student – Class**
- **Customer – Product**
- **Book – Author**



M:N

Many to Many

Student_Class

Student ID	Class ID	time
1	3	08:00
1	5	09:00
1	2	11:00
2	1	12:00
2	4	10:30
2	5	14:25
2	2	08:00

student

Student ID	Student name
1	John
2	Debbie

class

Class ID	Class name
1	English
2	Maths
3	Spanish
4	Biology
5	Science
6	Programming



1:1

One to One

- **Country – capital city**
- **Person – their fingerprints**
- **Email – user account**
- **User profile – user settings**



1:1

One to One

Country

id	name
1	France
2	Germany
3	Spain

Capital

id	name	country_id
1	Madrid	3
2	Berlin	2
3	Paris	1



1:N

One to Many

- Teacher – Course
- Country – City
- Customer – Order



1:N

One to Many

Teacher

Teacher_ID	Teacher_Name
1	Carmen
2	Veronica
3	Jorge

Course

Course_ID	Course_Name	Teacher_ID
1	Biology	1
2	Math	1
3	English	3

Design

Course_ID	Course_Name	Teacher_ID
1	Biology	1
2	Math	1
3	English	3

Field (ستون)

Record (سطر)



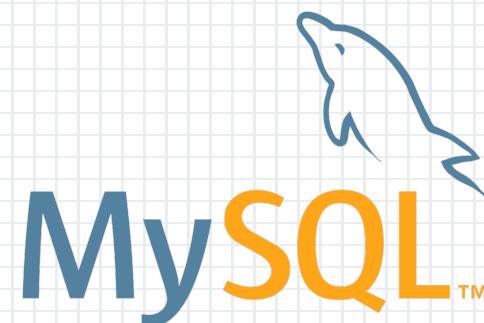
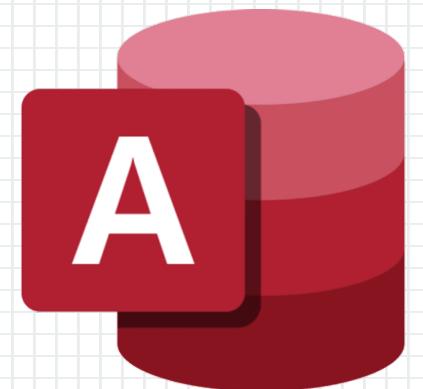
DBMS

DateBase Management System



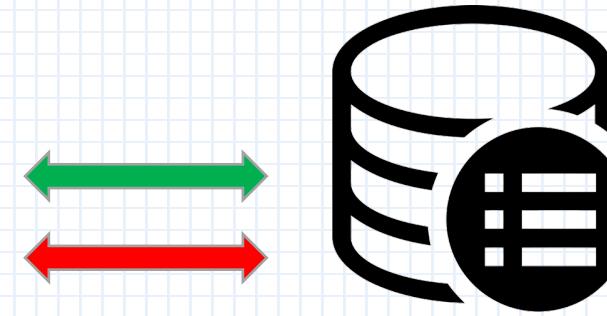
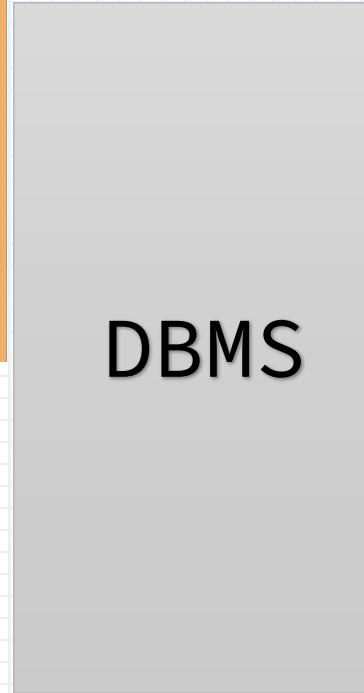
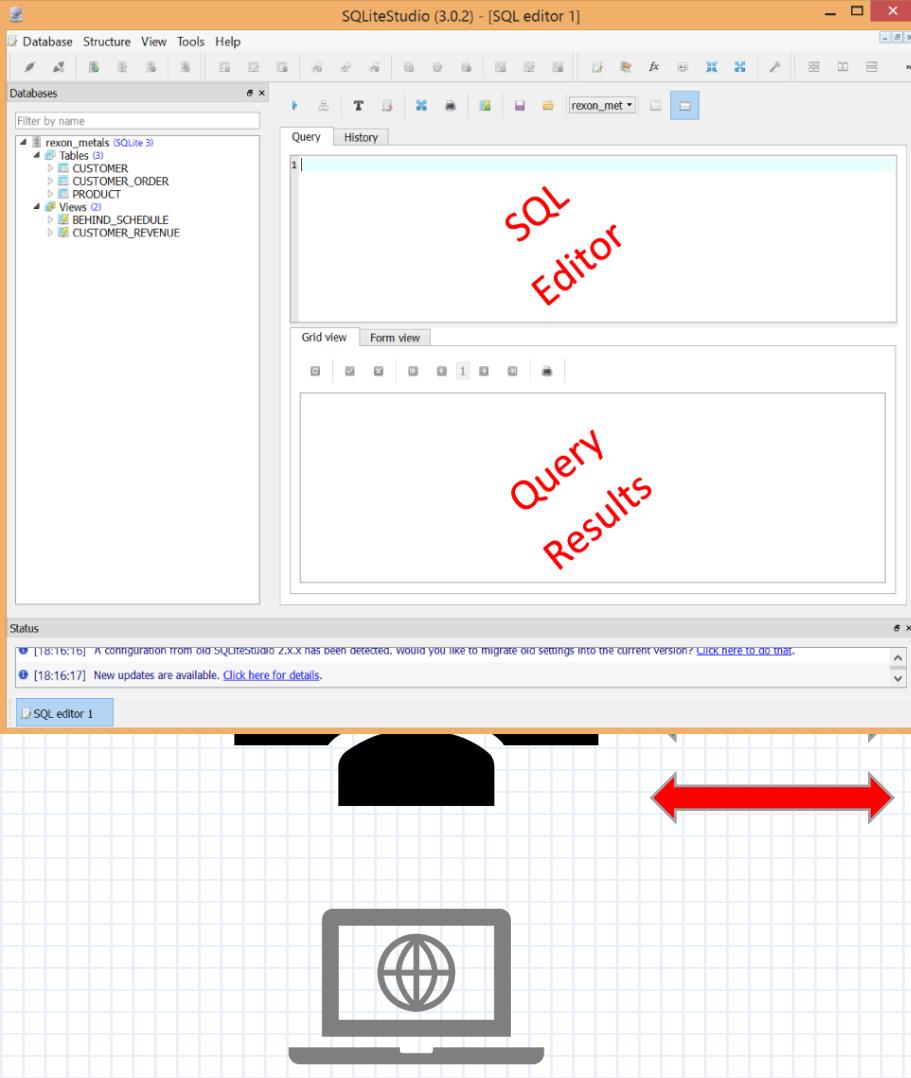
DBMS

سیستم مدیریت پایگاه داده برنامه ای است که عمل ذخیره سازی، بازیابی، امنیت و بهطور کل ارتباط با پایگاه داده را کنترل میکند. کاربران درخواست خود را به این نرم افزار ارسال میکنند و از طریق آن با پایگاه داده ارتباط برقرار میکنند.



ORACLE





DBMS



- SQLite is the most widely distributed database in the world. It is put on iPhones, iPads, Android devices, Windows phones, thermostats, car consoles, satellites, and ...
- But every technology has a trade-off. Because it has no server managing access to it, it fails in multiuser environments where multiple people can simultaneously edit the SQLite file. Still, for our training purposes, SQLite is perfect.

SQLiteStudio

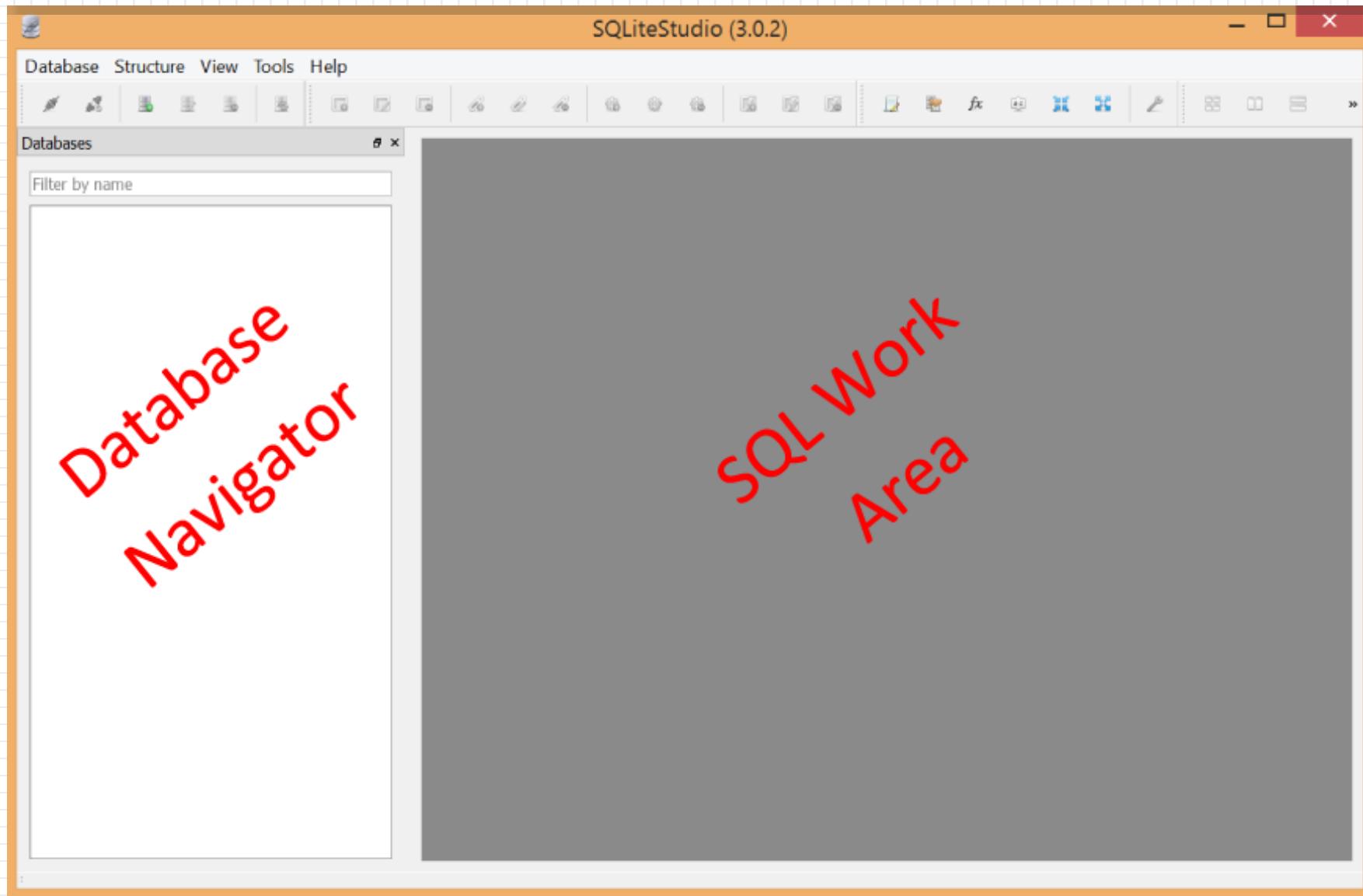
<http://sqlitestudio.pl/?act=download>

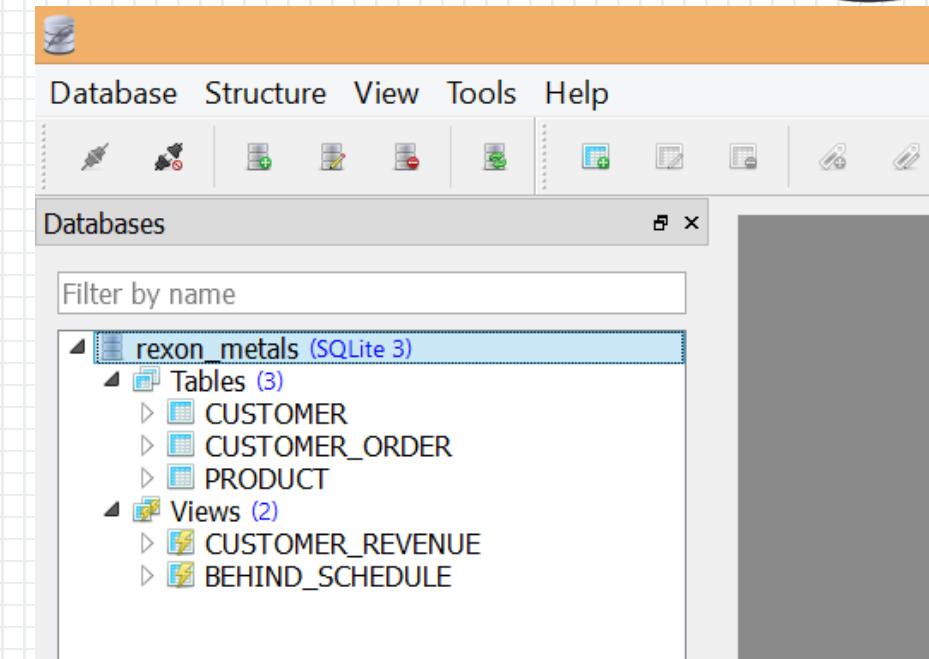
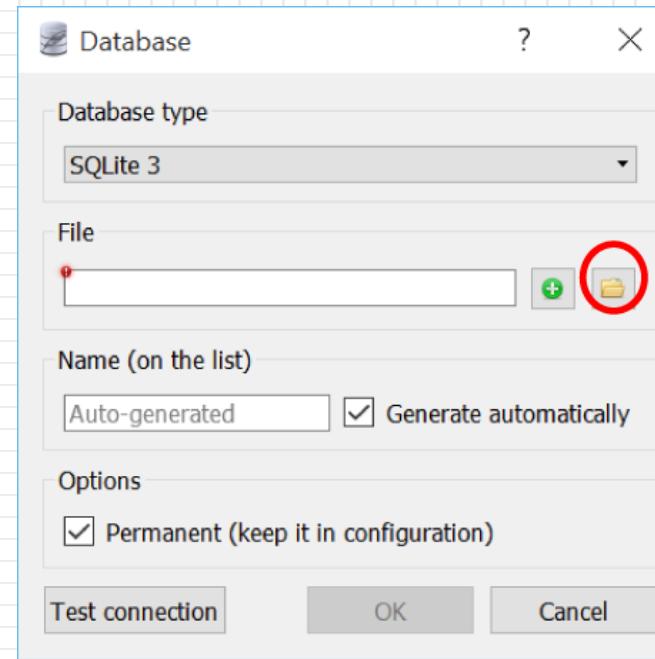
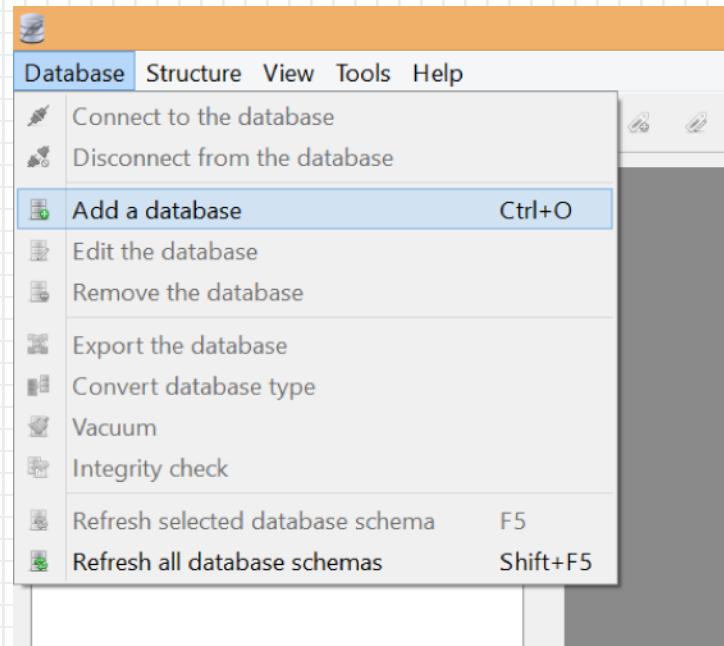




- **SQLiteStudio**
- <http://sqlitestudio.pl/?act=download>









SQLiteStudio (3.0.2)

Database Structure View Tools Help

Databases

Filter by name

rexon_metals (SQLite 3)

- Tables (3)
 - CUSTOMER
 - Columns (7)
 - CUSTOMER_ID
 - NAME
 - REGION
 - STREET_ADDRESS
 - CITY
 - STATE
 - ZIP
 - Indexes
 - Triggers
 - CUSTOMER_ORDER
 - PRODUCT
 - Views (2)
 - CUSTOMER_REVENUE
 - BEHIND_SCHEDULE

CUSTOMER (rexon_metals)

Structure Data Constraints Indexes Triggers DDL

Table name: CUSTOMER WITHOUT ROWID

	Name	Data type	P	F	U	H	N	C	Default value
1	CUSTOMER_ID	INTEGER	PK						NULL
2	NAME	TEXT							NULL
3	REGION	TEXT							NULL
4	STREET_ADDRESS	TEXT							NULL
5	CITY	TEXT							NULL
6	STATE	TEXT							NULL
7	ZIP	INTEGER							NULL

Type Name Details

SQLiteStudio (3.0.2)

Database Structure View Tools Help

Databases

Filter by name

rexon_metals (SQLite 3)

- Tables (3)
 - CUSTOMER
 - Columns (7)
 - CUSTOMER_ID
 - NAME
 - REGION
 - STREET_ADDRESS
 - CITY
 - STATE
 - ZIP
 - Indexes
 - Triggers
 - CUSTOMER_ORDER
 - PRODUCT
 - Views (2)
 - BEHIND_SCHEDULE
 - CUSTOMER_REVENUE

CUSTOMER (rexon_metals)

Structure Data Constraints Indexes Triggers DDL

Grid view Form view

	CUSTOMER_ID	NAME	REGION	STREET_ADDRESS	CITY	STATE	ZIP
1	1	LITE Industrial	Southwest	729 Ravine Way	Irving	TX	75014
2	2	Rex Tooling Inc	Southwest	6129 Collie Blvd	Dallas	TX	75201
3	3	Re-Barre Construction	Southwest	9043 Windy Dr	Irving	TX	75032
4	4	Prairie Construction	Southwest	264 Long Rd	Moore	OK	62104
5	5	Marsh Lane Metal Works	Southeast	9143 Marsh Ln	Avondale	LA	79782



SQL

Structured Query Language



SELECT

INSERT

DELETE

UPDATE

WHERE
AND
OR
NOT
IN

Order By
ASC
DESC

GROUP By

ANY
ALL

JOIN
LEFT JOIN
RIGHT JOIN
FULL JOIN

LIKE

HAVING

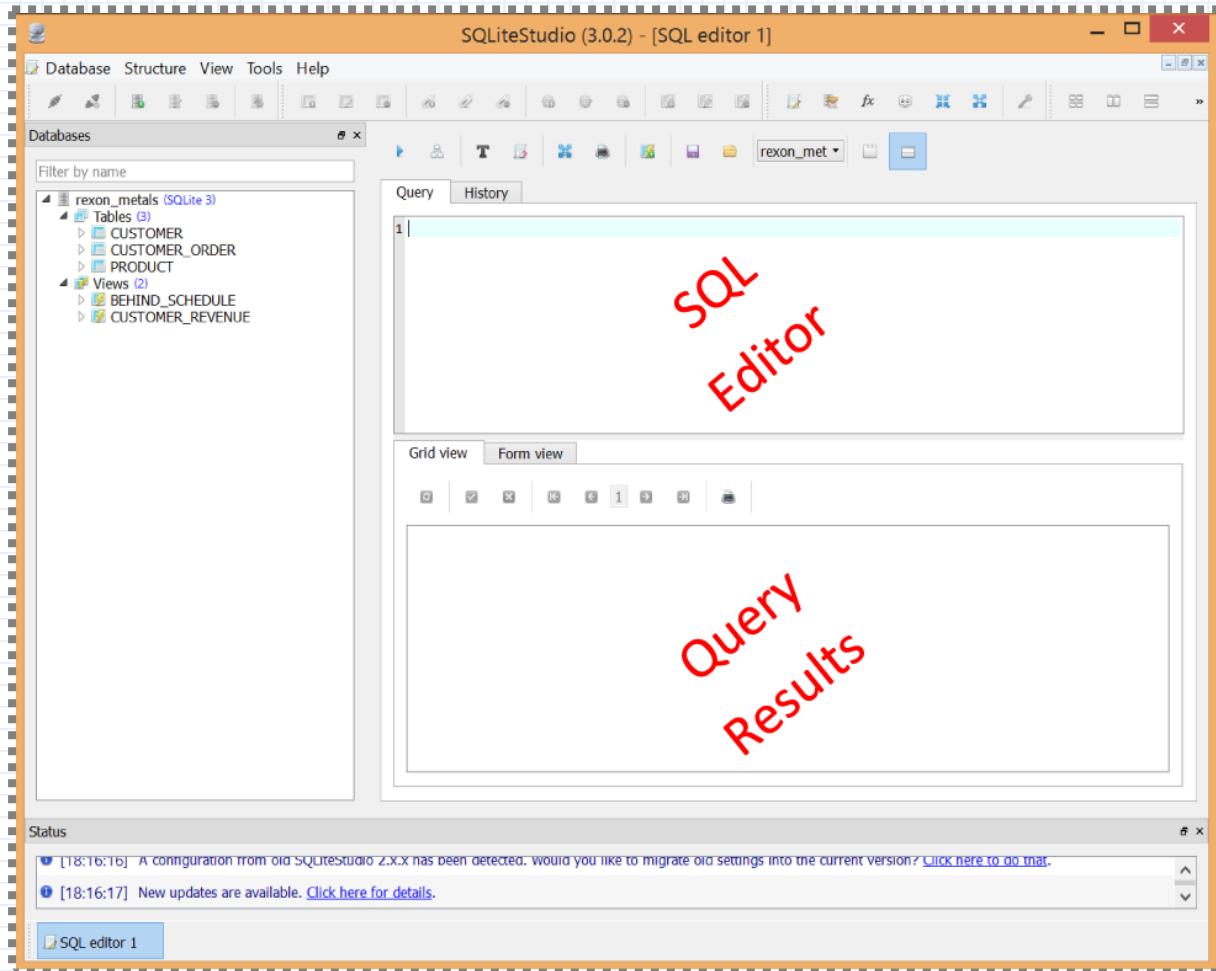
CASE

UNION



SELECT

Tools → Open SQL



SELECT

```
SELECT column1, column2, ...  
FROM table_name;
```

SELECT

```
SELECT *
FROM table_name;
```



SELECT DISTINCT

The SELECT DISTINCT statement is used to return only distinct (different) values.

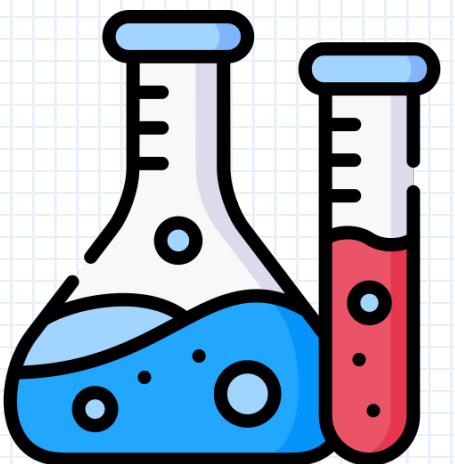
```
SELECT station_number  
FROM station_data
```

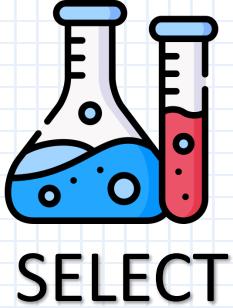
VS

```
SELECT DISTINCT station_number  
FROM station_data
```

SELECT

HANDs-ON LABs



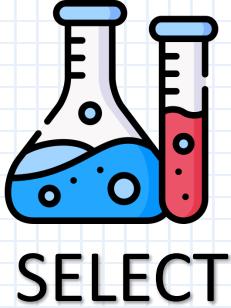


SELECT

1

کوئری بنویسید که تمام ستونهای جدول STATION_DATA را انتخاب کند.

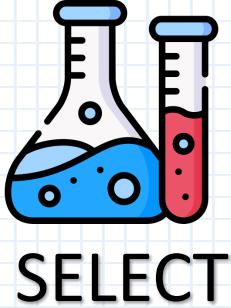




2

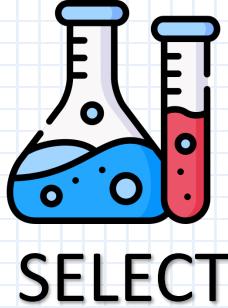
کوئی بنویسید کہ فقط ستون station_number را انتخاب کند.





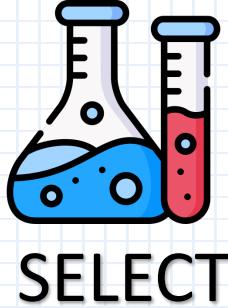
3

کوئری بنویسید که ستونهای station_number، temperature و report_code جدول STATION_DATA را انتخاب کند.



4

کوئری بنویسید که ستونهای station_number و temperature و report_code جدول STATION_DATA را انتخاب کند و نام ستون مربوط به دما را به dama تغییر دهد.

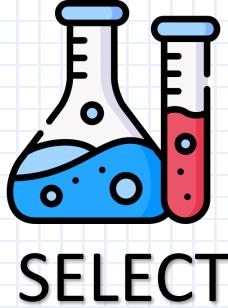


5

کوئری بنویسید که ستونهای station_number، temperature و report_code را انتخاب کند و یک ستون به اسم tempfa برای دما به فارنهایت اضافه کند.

```
SELECT
station_number,
report_code,
temperature,
temperature * 1.8 + 32 AS tempfa
FROM STATION_DATA
```



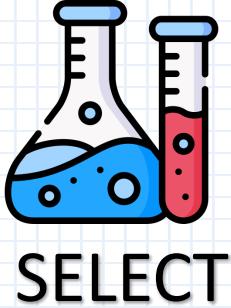


6

کوئری بنویسید که ستونهای station_number، temperature و report_code را انتخاب کند و یک ستون به اسم tempfa برای دما به فارنهایت اضافه کرده و آن را تا دو رقم اعشار رند کند.

```
SELECT
station_number,
report_code,
temperature,
round(temperature * 1.8 + 32, 2) AS tempfa
FROM STATION_DATA
```





کوئری بنویسید که ستونهای روز، ماه و سال را در ستونی به نام date نمایش دهد.

7

WHERE

Filter records:

```
SELECT column1, column2, ...
FROM table_name
WHERE condition
```

WHERE

```
SELECT *  
FROM station_data  
WHERE year = 2010;
```

```
SELECT *  
FROM station_data  
WHERE year != 2010
```

```
SELECT *  
FROM station_data  
WHERE year BETWEEN 2005 and 2010
```

```
SELECT *  
FROM station_data  
WHERE year > 2005  
AND  
year < 2010
```

```
SELECT *  
FROM station_data  
WHERE MONTH = 3  
OR  
MONTH=6
```

```
SELECT *  
FROM station_data  
WHERE  
MONTH IN (3,6,9,12)
```

```
SELECT *  
FROM station_data  
WHERE report_code = '513A63'
```

```
SELECT *  
FROM station_data  
WHERE  
MONTH NOT IN (3,6,9,12)
```

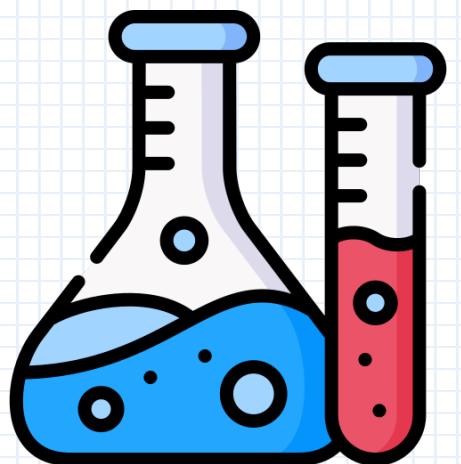
```
SELECT *  
FROM station_data  
WHERE  
report_code LIKE 'A%'
```

```
SELECT *  
FROM station_data  
WHERE  
length(report_code) != 6
```



WHERE

HANDs-ON LABs

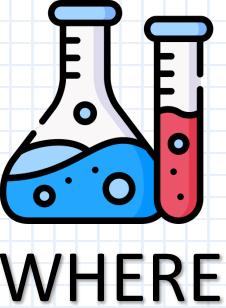




8

WHERE

کوئی بنویسید که تمام اطلاعات مربوط به سال ۲۰۱۰ را نمایش دهد.

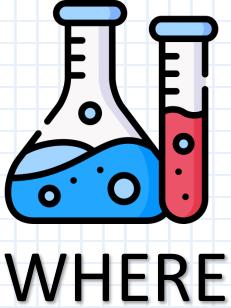


9

WHERE

کوئی بنویسید که تمام اطلاعات مربوط به جز سال ۲۰۱۰ را نمایش دهد.



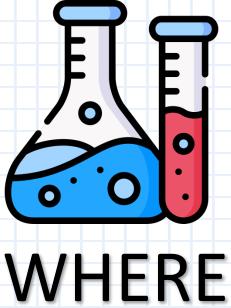


10

WHERE

کوئی بنویسید که تمام اطلاعات مربوط به بین سال ۲۰۰۴ و ۲۰۰۹ را نمایش دهد.



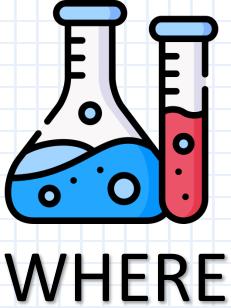


11

WHERE

کوئی بنویسید که تمام اطلاعات مربوط به پس از سال ۲۰۰۸ و قبل از سال ۲۰۰۶ را نمایش دهد.



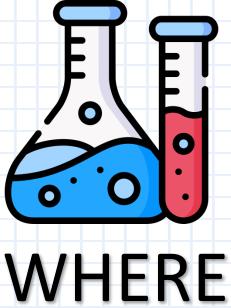


12

WHERE

کوئی بنویسید که تمام اطلاعات مربوط به ماه اول یا دوم یا سوم سال ۲۰۱۰ را نمایش دهد.





13

WHERE

کوئی بنویسید که تمام اطلاعات مربوط به شش ماه اول
هر سال را نمایش دهد.



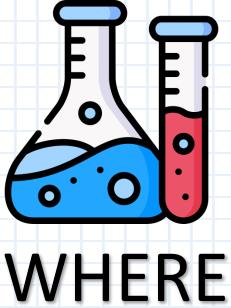


14

WHERE

کوئی بنویسید که تمام اطلاعات مربوط به شش ماه اول
هر سال به جز ۵ روز پایانی هر ماه را نمایش دهد.





15

WHERE

کوئری بنویسید که تمام اطلاعات مربوط به اولین روز از اولین ماه هر فصل سالهای ۲۰۰۰ تا ۲۰۰۵ را نمایش دهد.



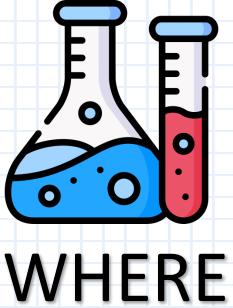


16

WHERE

کوئری بنویسید که تمام اطلاعاتی که آنها report_code ۶ حرفی است را نمایش دهد.



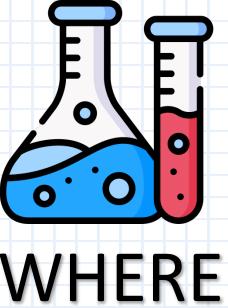


17

WHERE

کوئری بنویسید که تمام اطلاعاتی که report_code آنها با A شروع می‌شود را نمایش دهد.



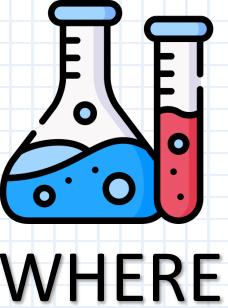


18

WHERE

کوئری بنویسید که تمام اطلاعاتی که report_code آنها با A تمام می‌شود را نمایش دهد.



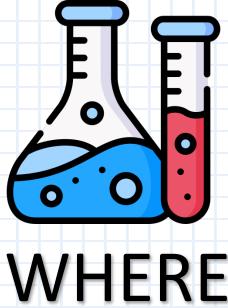


19

WHERE

کوئری بنویسید که تمام اطلاعاتی که report_code آنها در دومین حرف A دارد را نمایش دهد.



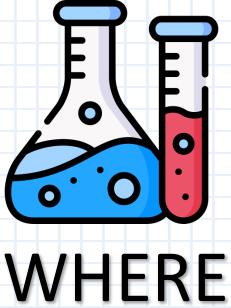


20

WHERE

کوئری بنویسید که تمام اطلاعاتی که report_code در دومین حرف A دارد و با A پایان می‌یابد را نمایش دهد.



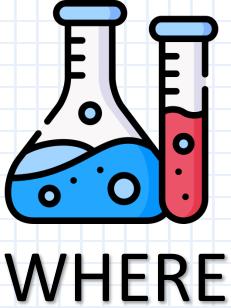


21

WHERE

کوئی بنویسید که تمام اطلاعاتی که بارانی بوده و طوفانی نبوده را نمایش دهد.





22

WHERE

کوئری بنویسید که تمام اطلاعاتی که precipitation نهایا نیست را نمایش دهد.



INSERT INTO

```
INSERT INTO table_name (column1, column2,...)  
VALUES (value1, value2,...);
```

```
INSERT INTO table_name  
VALUES (value1, value2,...);
```



INSERT INTO

```
INSERT INTO Customers (CustomerName, City, Country)
VALUES ('ALI', 'TEHRAN', 'IRAN');
```

```
INSERT INTO Customers (CustomerName, City, Country)
VALUES
('ALI1', 'TEHRAN', 'IRAN'),
('ALI2', 'TEHRAN', 'IRAN'),
('ALI3', 'TEHRAN', 'IRAN'),
('ALI4', 'TEHRAN', 'IRAN');
```



UPDATE

```
UPDATE table_name
      SET column1 = value1, column2 = value2, ...
      WHERE condition;
```

UPDATE

```
UPDATE Customers  
SET ContactName = 'Alfred Schmidt', City= 'Frankfurt'  
WHERE CustomerID = 1;
```



DELETE

```
DELETE  
FROM table_name  
WHERE condition;
```

DELETE

```
DELETE FROM Customers  
WHERE CustomerName='Alfreds Futterkiste';
```



GROUP BY

- The GROUP BY statement groups rows that have the same values into summary rows, like "find the number of customers in each country".
- The GROUP BY statement is often used with aggregate functions (COUNT(), MAX(), MIN(), SUM(), AVG()) to group the result-set by one or more columns.



GROUP BY

Aggregating data:

```
SELECT column1, column2, ...
FROM table_name
GROUP BY column1, column2, ...
```

GROUP BY

```
SELECT COUNT(*) AS record_count  
FROM station_data;
```

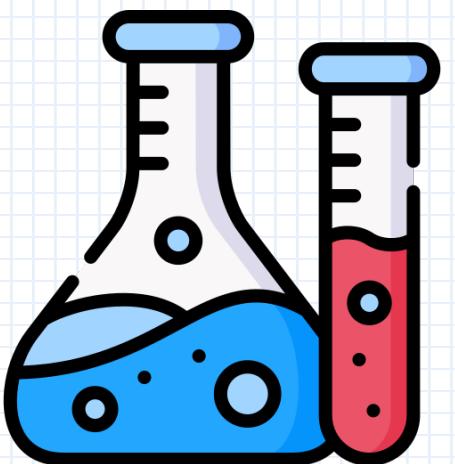
```
SELECT COUNT(*) AS record_count  
FROM station_data  
WHERE tornado = 1;
```

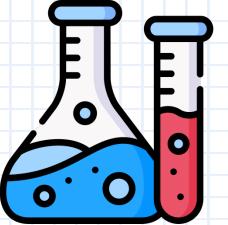
```
SELECT year, COUNT(*) AS record_count  
FROM station_data  
WHERE tornado = 1  
GROUP BY year;
```



GROUP BY

HANDs-ON LABs



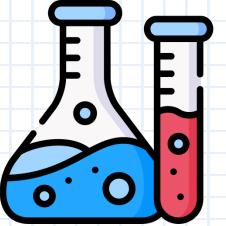


GROUP BY

کوئی بنویسید که تعداد تمام گربادهایی که در هر سال اتفاق افتاده است را به تفکیک سال نمایش دهد.

23



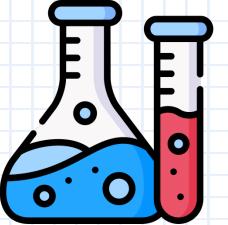


GROUP BY

24

کوئی بنویسید که تعداد تمام گربادهایی که در هر سال و ماه اتفاق افتاده است را به تفکیک سال و ماه نمایش دهد.





GROUP BY

25

کوئری بنویسید که تعداد روزهایی که دما از ۲۵ بیشتر بوده را به تفکیک ماه و سال نمایش دهد.



ORDER BY

- The ORDER BY keyword is used to sort the result-set in ascending or descending order.
- ascending order by default. To sort the records in descending order, use the DESC keyword.



ORDER BY

sort data:

```
SELECT column1, column2, ...
FROM table_name
ORDER BY column1, column2, ... ASC|DESC;
```

ORDER BY

```
SELECT *  
FROM station_data  
ORDER BY year;
```

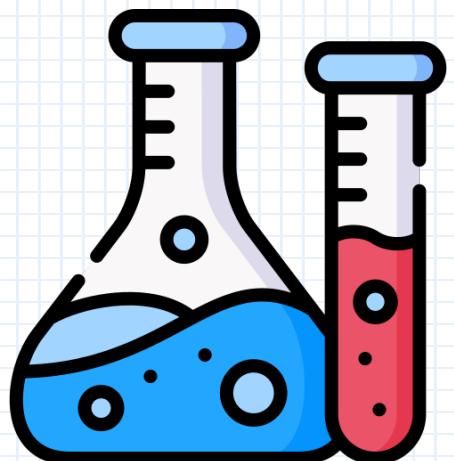
```
SELECT *  
FROM station_data  
ORDER BY year, month;
```

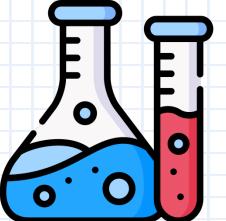
```
SELECT *  
FROM station_data  
ORDER BY year DESC, month;
```



ORDER BY

HANDs-ON LABs



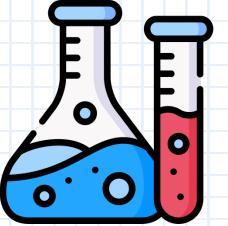


ORDER BY

26

کوئی بنویسید که تمام اطلاعات جدول را از پایین‌ترین دما تا بالاترین دما به صورت مرتب نمایش دهد.



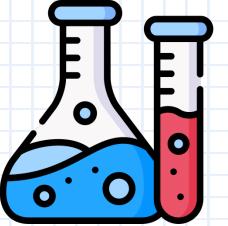


ORDER BY

27

کوئی بنویسید که تمام اطلاعات جدول را از بالاترین دما تا پایین‌ترین دما به صورت مرتب نمایش دهد.



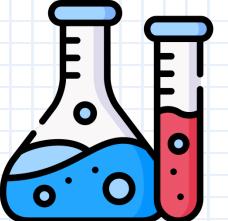


ORDER BY

28

کوئی بنویسید که تمام اطلاعات جدول را از بالاترین دما تا پایین‌ترین دما و از سالهای کمتر به بیشتر به صورت مرتب نمایش دهد.



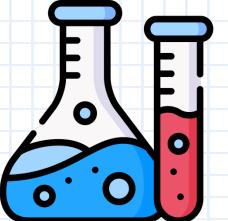


ORDER BY

29

کوئی بنویسید که تمام اطلاعات جدول را از روز، ماه و سال بیشتر به کمتر به صورت مرتب نمایش دهد.





ORDER BY

30

کوئی بنویسید که تعداد تمام گربادهایی که در هر سال و ماه اتفاق افتاده است را به تفکیک سال و ماه و به صورت نزولی نمایش دهد.



never include null values in their calculations.

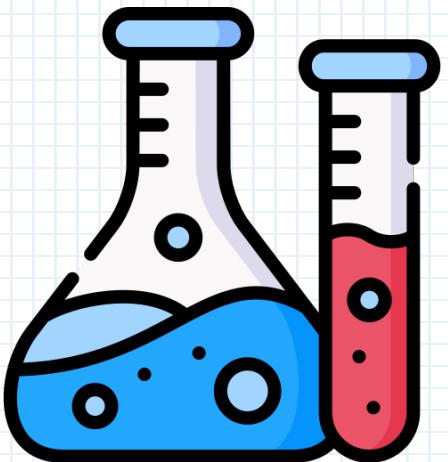
Aggregate Function

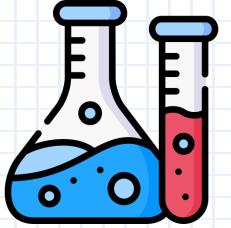
MIN()	It is used to extract the minimum value inserted in the specified column
MAX()	It is used to extract the maximum value inserted in the specified column
AVG()	It is used to extract the average value of all the values inserted in the specified column
COUNT()	It is used to count the total entries of rows of the specified column
SUM()	It is used to find out the sum of all the values of the specified column
UPPER()	It is used to convert all the string values to the upper case of the specified column
LOWER()	It is used to convert all the string values to the lower case of the specified column
LENGTH()	It is used to find the number of characters or letters the in a specified string
ABS()	It will return the absolute values of the specified column



Aggregate functions

HANDS-ON LABS





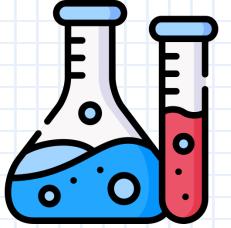
Aggregate

31

کوئی بنویسید که کمترین دمای فصل بهار در سال ۲۰۰۹ را نمایش دهد.

```
SELECT MIN(temperature)  
FROM station_data
```



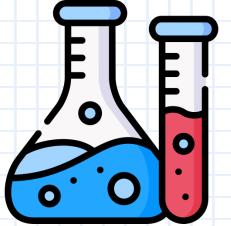


Aggregate

32

کوئی بنویسید که بیشترین دمای فصل بهار در سال ۲۰۰۹ را نمایش دهد.



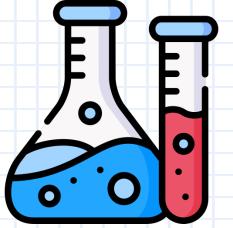


Aggregate

کوئری بنویسید که معدل دما را در هر ماه سال ۲۰۰۹ را نمایش دهد.

33



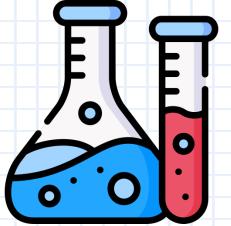


Aggregate

34

کوئی بنویسید که تعداد دفعاتی که باران باریده است را در هر سال نمایش دهد.



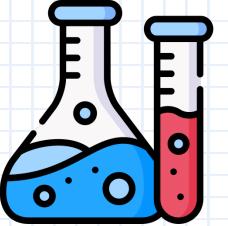


Aggregate

کوئی بنویسید که جمع میزان بارش برف را در هر سال
نمایش دهد.

35



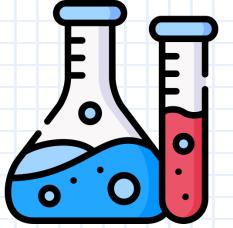


Aggregate

36

کوئی بنویسید که جمع میزان بارش برف، جمع بارش و حداقل میزان بارش را از سال ۲۰۰۰ به بعد نمایش دهد.





Aggregate

کوئی بنویسید که جمع میزان بارش در هر سال را زمانی که گردباد بوده است را نمایش دهد.

37



HAVING

The HAVING clause was added to SQL because the WHERE keyword cannot be used with aggregate functions.

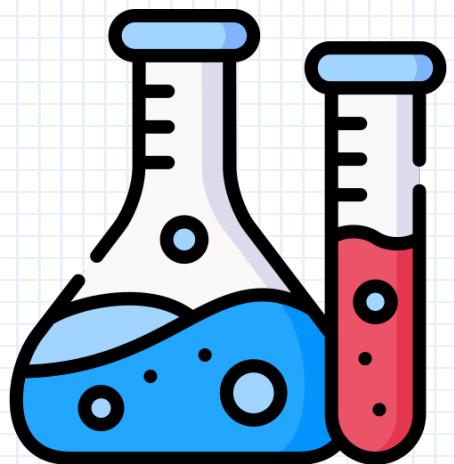


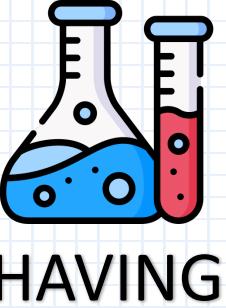
```
SELECT year,  
SUM(precipitation) as total_precipitation  
FROM station_data  
WHERE total_precipitation > 30  
GROUP BY year
```



```
SELECT year,  
SUM(precipitation) as total_precipitation  
FROM station_data  
GROUP BY year  
HAVING total_precipitation > 30
```

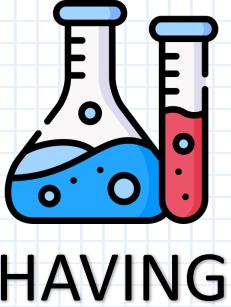
HAVING HANDs-ON LABs





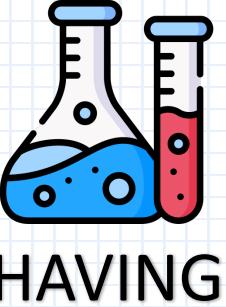
کوئری بنویسید که جمع میزان بارش در هر سال را زمانی اگر بیش از 2 بوده است را نمایش دهد.

38



کوئری بنویسید که جمع میزان برف در هر ماه سال ۲۰۰۹ را زمانی اگر بیش از ۵ بوده است را نمایش دهد.

39



40

کوئی بنویسید که جمع میزان بارش در هر سال را زمانی که گردباد بوده اگر بیش از ۳ بوده است را نمایش دهد.



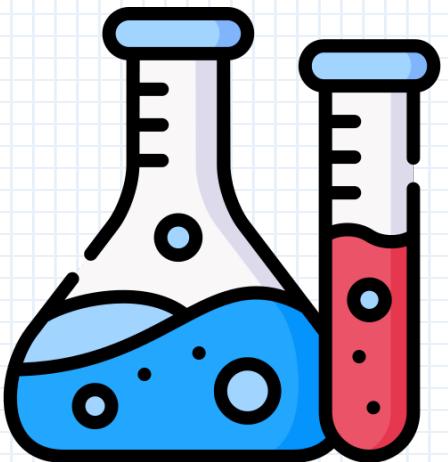
CASE

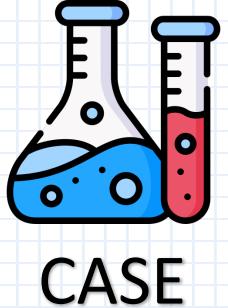
A CASE statement allows us to map one or more conditions to a corresponding value for each condition.

```
CASE
    WHEN condition1 THEN result1
    WHEN condition2 THEN result2
    WHEN conditionN THEN resultN
    ELSE result
END;
```

CASE

HANDs-ON LABs

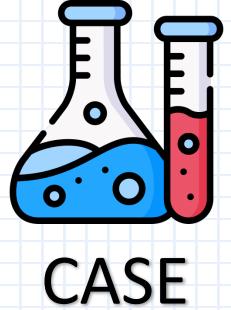




41

کوئی بنویسید که ستون کد، سال، ماه، روز، سرعت باد و شدت باد را اگر بیشتر از ۴۰ بود، high اگر بین ۳۰ تا ۴۰ بود moderate و اگر زیر ۳۰ بود low نمایش دهد.

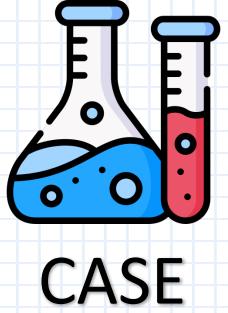
```
SELECT report_code, year, month, day, wind_speed,  
CASE  
    WHEN wind_speed >= 40 THEN 'HIGH'  
    WHEN wind_speed >= 30 AND wind_speed < 40 THEN 'MODERATE'  
    ELSE 'LOW'  
END as wind_severity  
FROM station_data
```



42

```
SELECT year, month,  
SUM(CASE WHEN tornado = 1 THEN precipitation ELSE 0 END) as tornado_precipitation,  
SUM(CASE WHEN tornado = 0 THEN precipitation ELSE 0 END) as non_tornado_precipitation  
FROM station_data  
GROUP BY year, month
```

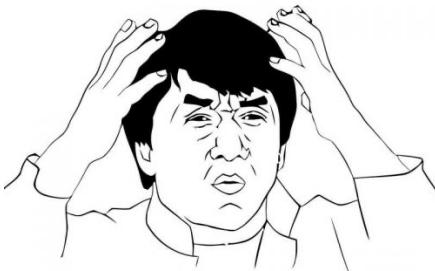




CASE

43

```
SELECT month,  
AVG(CASE WHEN rain OR hail THEN temperature ELSE null END)  
AS avg_precipitation_temp,  
AVG(CASE WHEN NOT (rain OR hail) THEN temperature ELSE null END)  
AS avg_non_precipitation_temp  
FROM STATION_DATA  
WHERE year > 2000  
GROUP BY month
```



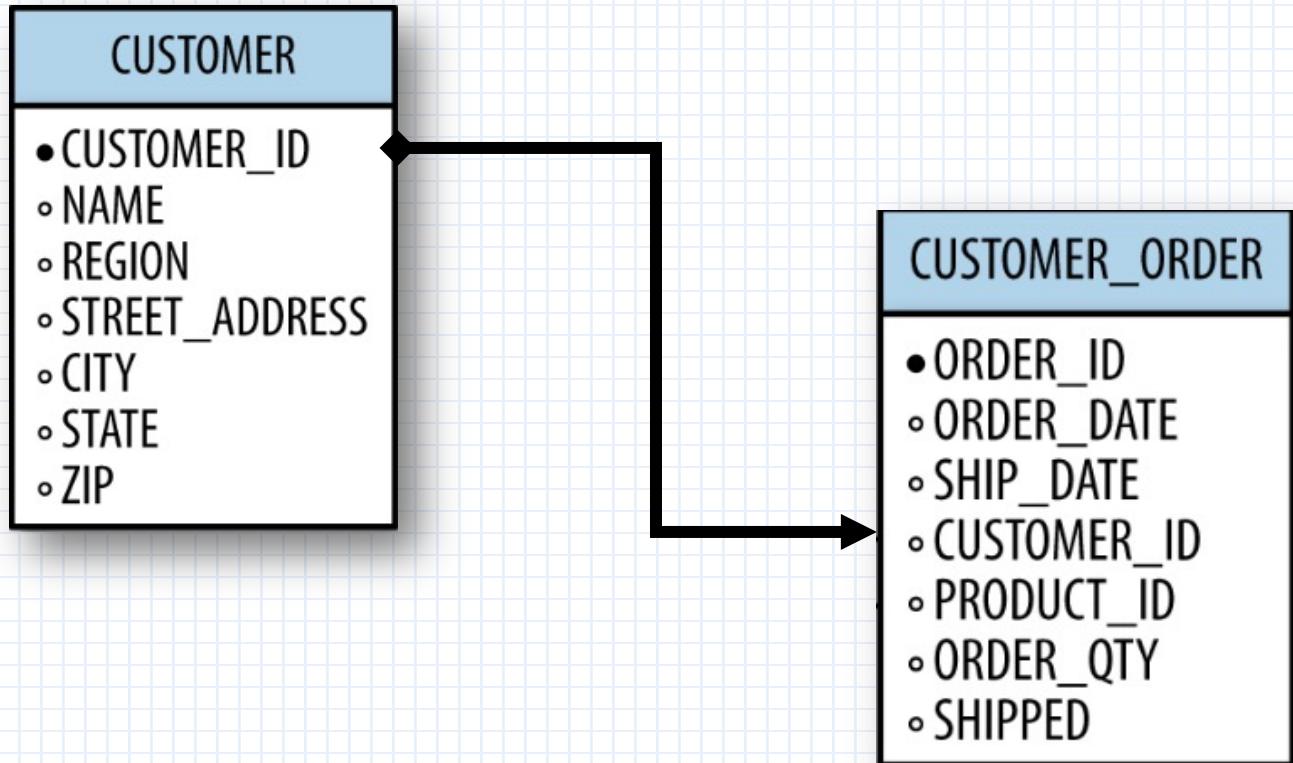
JOIN

A **JOIN** clause is used to combine rows from two or more tables, based on a related column between them.

- Inner Join
- Left Join
- Right Join
- Full Join
- Self Join



JOIN



JOIN

CUSTOMER

CUSTOMER_ID	NAME	REGION	STREET ADDRESS	CITY	STATE	ZIP
1	LITE Industrial	Southwest	729 Ravine Way	Irving	TX	75014
2	Rex Tooling Inc	Southwest	6129 Collie Blvd	Dallas	TX	75201
3	Re-Barre Construction	Southwest	9043 Windy Dr	Irving	TX	75032
4	Prairie Construction	Southwest	264 Long Rd	Moore	OK	62104
5	Marsh Lane Metal Works	Southeast	9143 Marsh Ln	Avondale	LA	79782

CUSTOMER_ORDER

ORDER_ID	ORDER_DATE	SHIP_DATE	CUSTOMER_ID	PRODUCT_ID	ORDER_QTY	SHIPPED
1	2015-04-20	2015-04-23	3	5	300	false
2	2015-04-18	2015-04-22	5	4	375	false
3	2015-04-15	2015-04-18	1	1	450	false
4	2015-04-17	2015-04-20	3	2	500	false
5	2015-04-18	2015-04-21	3	2	600	false



CUSTOMER

CUSTOMER_ID	NAME	REGION	STREET ADDRESS	CITY	STATE	ZIP
1	LITE Industrial	Southwest	729 Ravine Way	Irving	TX	75014
2	Rex Tooling Inc	Southwest	6129 Collie Blvd	Dallas	TX	75201
3	Re-Barre Construction	Southwest	9043 Windy Dr	Irving	TX	75032
4	Prairie Construction	Southwest	264 Long Rd	Moore	OK	62104
5	March Lane Metal Works	Southeast	9143 Marsh Ln	Avondale	LA	79782

CUSTOMER_ORDER

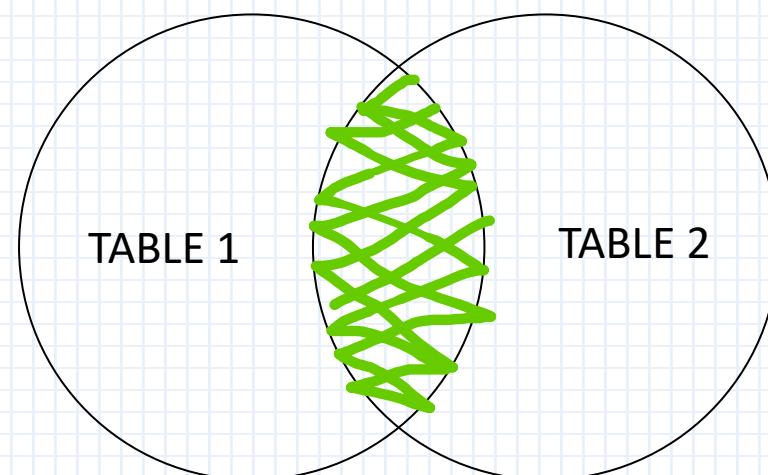
ORDER_ID	ORDER_DATE	SHIP_DATE	CUSTOMER_ID	PRODUCT_ID	ORDER_QTY	SHIPPED
1	2015-04-20	2015-04-23	3	5	300	false
2	2015-04-18	2015-04-22	5	4	375	false
3	2015-04-15	2015-04-18	1	1	450	false
4	2015-04-17	2015-04-20	3	2	500	false
5	2015-04-18	2015-04-21	3	2	600	false



INNER JOIN

The **INNER JOIN** keyword selects records that have matching values in both tables.

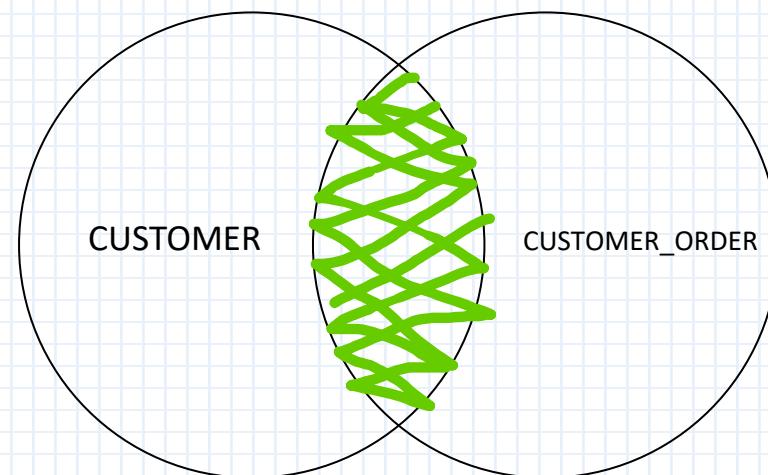
```
SELECT column_names  
FROM table1  
INNER JOIN table2  
ON table1.column_name = table2.column_name;
```



INNER JOIN

- If an CUSTOMER does not have a corresponding ORDER or if a ORDER does not have any CUSTOMER, they will not appear in the result set.

```
SELECT *
FROM CUSTOMER
INNER JOIN CUSTOMER_ORDER
ON CUSTOMER.CUSTOMER_ID = CUSTOMER_ORDER.CUSTOMER_ID
```



INNER JOIN

CUSTOMER

CUSTOMER ID	NAME	REGION	STREET ADDRESS	CITY	STATE	ZIP
1	LITE Industrial	Southwest	729 Ravine Way	Irving	TX	75014
2	Rex Tooling Inc	Southwest	6129 Collie Blvd	Dallas	TX	75201
3	Re-Barre Construction	Southwest	9043 Windy Dr	Irving	TX	75032
4	Prairie Construction	Southwest	264 Long Rd	Moore	OK	62104
5	Marsh Lane Metal Works	Southeast	9143 Marsh Ln	Avondale	LA	79782

CUSTOMER_ORDER

ORDER ID	ORDER DATE	SHIP DATE	CUSTOMER ID	PRODUCT ID	ORDER QTY	SHIPPED
1	2015-05-15	2015-05-18	1		1	450 false
2	2015-05-18	2015-05-21	3		2	600 false
3	2015-05-20	2015-05-23	3		5	300 false
4	2015-05-18	2015-05-22	5		4	375 false
5	2015-05-17	2015-05-20	3		2	500 false

INNER JOIN

INNER JOINED

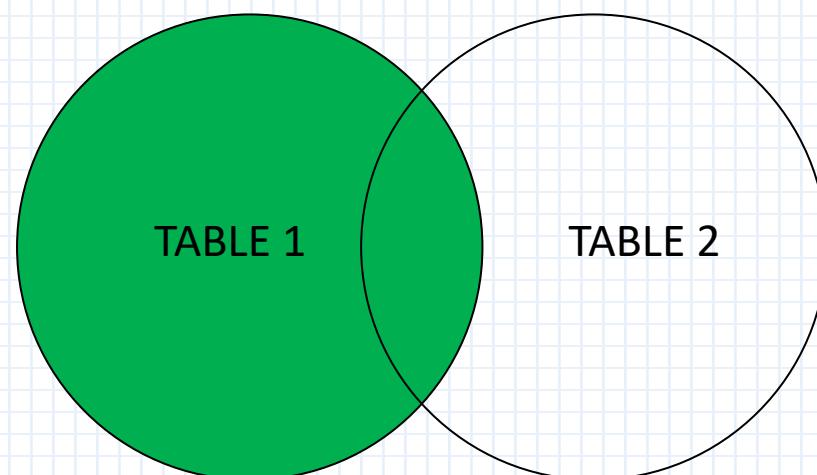
ORDER ID	CUSTOMER ID	ORDER DATE	SHIP DATE	NAME	STREET ADDRESS	CITY	STATE	ZIP	PRODUCT ID	ORDER QTY
1	1	2015-05-15	2015-05-18	LITE Industrial	729 Ravine Way	Irving	TX	75014	1	450
2	3	2015-05-18	2015-05-21	Re-Barre Construction	9043 Windy Dr	Irving	TX	75032	2	600
3	3	2015-05-20	2015-05-23	Re-Barre Construction	9043 Windy Dr	Irving	TX	75032	5	300
4	5	2015-05-18	2015-05-22	Marsh Lane Metal Works	9143 Marsh Ln	Avondale	LA	79782	4	375
5	3	2015-05-17	2015-05-20	Re-Barre Construction	9043 Windy Dr	Irving	TX	75032	2	500



LEFT JOIN

The **LEFT JOIN** keyword returns all records from the left table (table1), and the matching records from the right table (table2).

```
SELECT column_names  
FROM table1  
LEFT JOIN table2  
ON table1.column_name = table2.column_name;
```



LEFT JOIN

CUSTOMER

CUSTOMER ID	NAME	REGION	STREET ADDRESS	CITY	STATE	ZIP
1	LITE Industrial	Southwest	729 Ravine Way	Irving	TX	75014
2	Rex Tooling Inc	Southwest	6129 Collie Blvd	Dallas	TX	75201
3	Re-Barre Construction	Southwest	9043 Windy Dr	Irving	TX	75032
4	Prairie Construction	Southwest	264 Long Rd	Moore	OK	62104
5	Marsh Lane Metal Works	Southeast	9143 Marsh Ln	Avondale	LA	79782

CUSTOMER_ORDER

ORDER ID	ORDER DATE	SHIP DATE	CUSTOMER ID	PRODUCT ID	ORDER QTY	SHIPPED
1	2015-05-15	2015-05-18	1		1	false
2	2015-05-18	2015-05-21	3		2	false
3	2015-05-20	2015-05-23	3		5	false
4	2015-05-18	2015-05-22	5		4	false
5	2015-05-17	2015-05-20	3		2	false

NULL



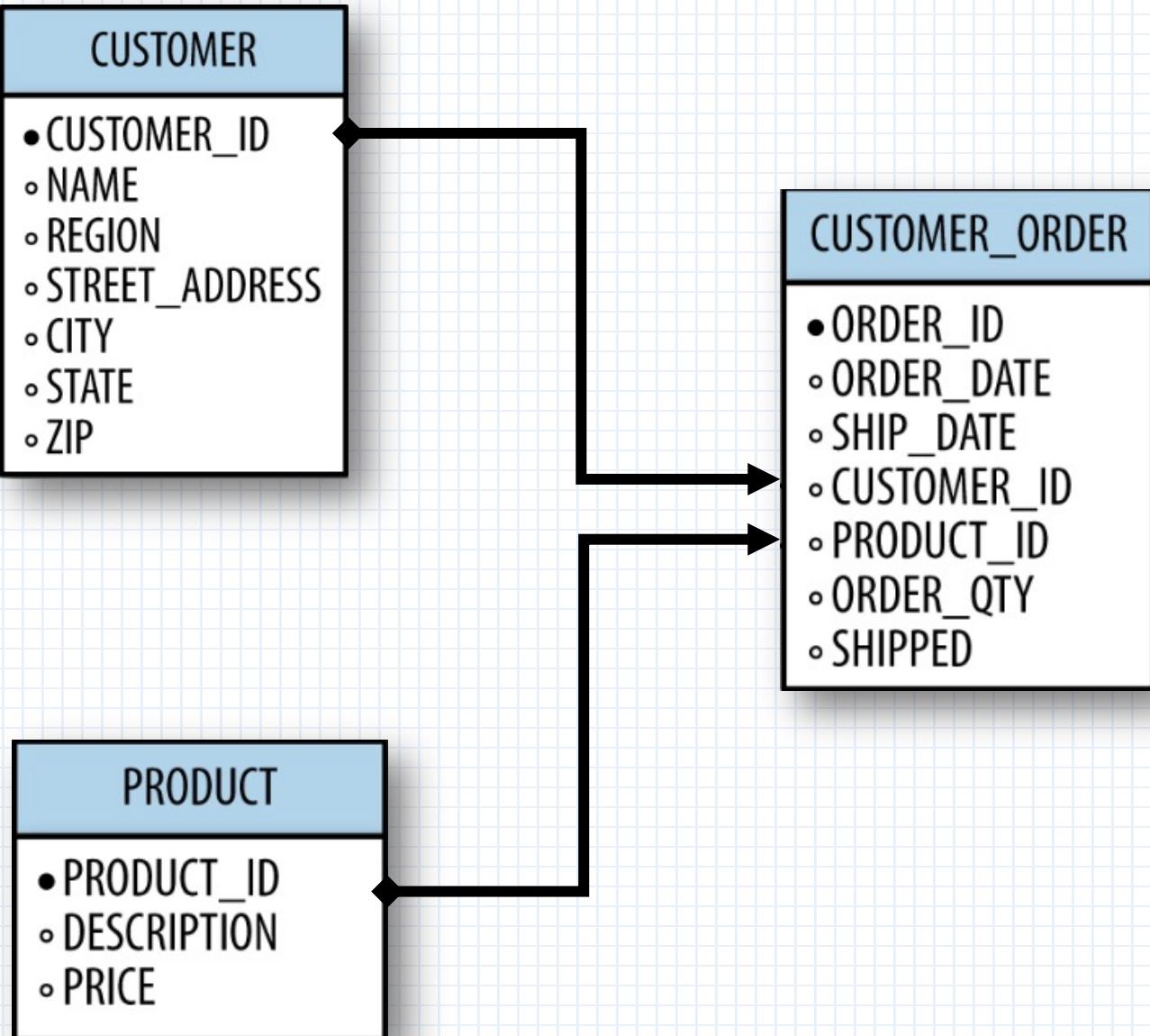
LEFT JOIN

LEFT OUTER JOINED

CUSTOMER ID	NAME	STREET ADDRESS	CITY	STATE	ZIP	ORDER DATE	SHIP DATE	ORDER ID	PRODUCT ID	ORDER QTY
1	LITE Industrial	729 Ravine Way	Irving	TX	75014	2015-05-15	2015-05-18	1	1	450
2	Rex Tooling Inc	6129 Collie Blvd	Dallas	TX	75201	NULL	NULL	NULL	NULL	NULL
3	Re-Barre Construction	9043 Windy Dr	Irving	TX	75032	2015-05-17	2015-05-20	5	2	500
3	Re-Barre Construction	9043 Windy Dr	Irving	TX	75032	2015-05-18	2015-05-21	2	2	600
3	Re-Barre Construction	9043 Windy Dr	Irving	TX	75032	2015-05-20	2015-05-23	3	5	300
4	Prairie Construction	264 Long Rd	Moore	OK	62104	NULL	NULL	NULL	NULL	NULL
5	Marsh Lane Metal Works	9143 Marsh Ln	Avondale	LA	79782	2015-05-18	2015-05-22	4	4	375



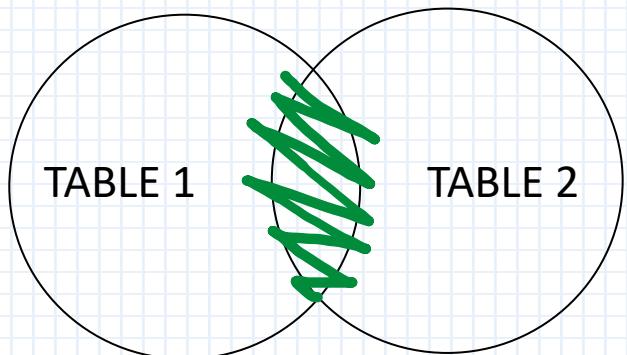
JOINING MULTIPLE TABLE



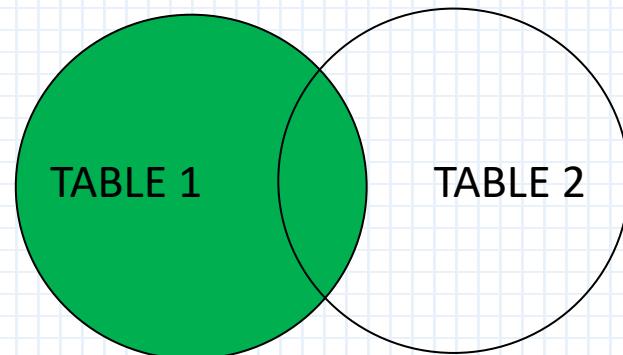
JOINING MULTIPLE TABLE

```
SELECT
    ORDER_ID,
    CUSTOMER.CUSTOMER_ID,
    NAME AS CUSTOMER_NAME,
    STREET_ADDRESS,
    CITY,
    STATE,
    ZIP,
    ORDER_DATE, PRODUCT_ID, DESCRIPTION, ORDER_QTY
FROM CUSTOMER
INNER JOIN CUSTOMER_ORDER
ON CUSTOMER_ORDER.CUSTOMER_ID = CUSTOMER.CUSTOMER_ID
INNER JOIN PRODUCT
ON CUSTOMER_ORDER.PRODUCT_ID = PRODUCT.PRODUCT_ID
```

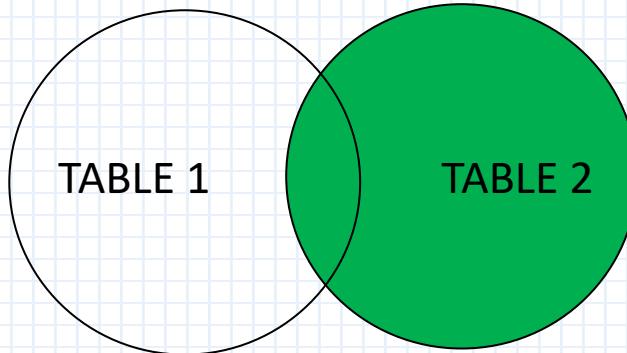




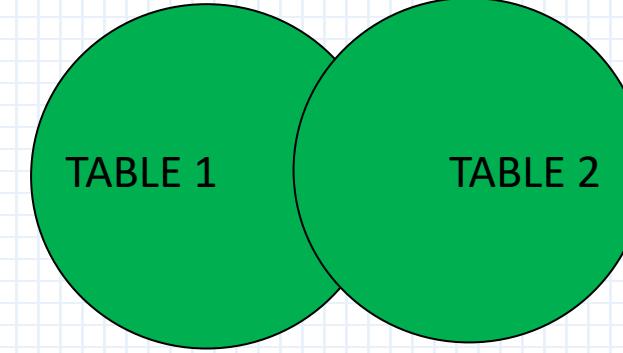
INNER JOIN



LEFT JOIN



RIGHT JOIN



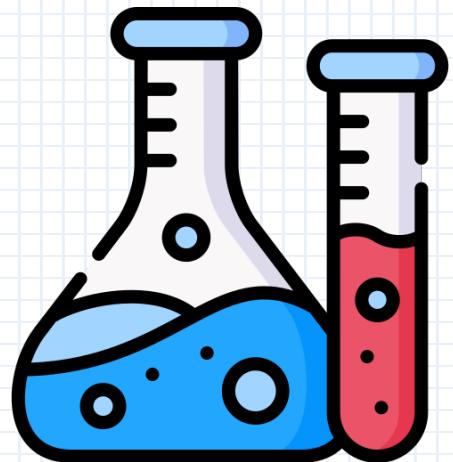
FULL JOIN

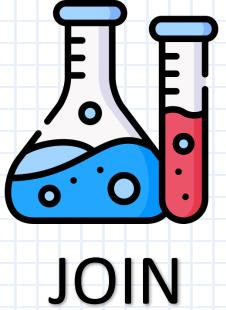
JOIN



JOIN

HANDs-ON LABs

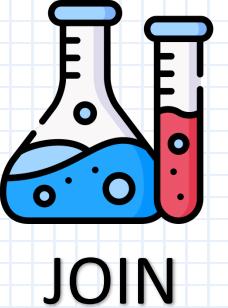




44

کوئری بنویسید که تمام ستونهای دو جدول CUSTOMER و CUSTOMER_ORDER که بهم مرتبط هستند را در یک جدول نمایش دهد.

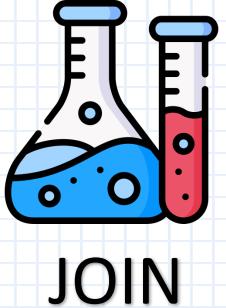




45

کوئری بنویسید که آیدی، اسم، شهر، آدرس مشتریان و تاریخ سفارش آنها را در یک جدول نمایش دهد.

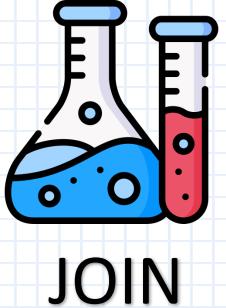
```
SELECT CUSTOMER.CUSTOMER_ID, NAME,  
STREET_ADDRESS,  
CITY,  
ORDER_DATE  
FROM CUSTOMER  
INNER JOIN CUSTOMER_ORDER  
ON CUSTOMER.CUSTOMER_ID = CUSTOMER_ORDER.CUSTOMER_ID
```



46

کوئی بنویسید که مشخصات تمام محصولاتی که توسط مشتری با آیدی ۳ سفارش داده است را نمایش دهد.

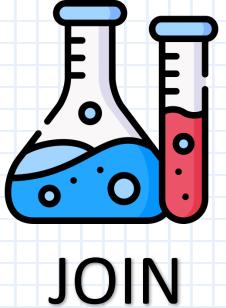




کوئری بنویسید آیدی، اسم و آدرس تمام مشتریانی که تا
کنون سفارشی ثبت نکرده‌اند را نمایش دهد.

47

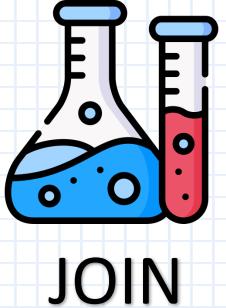




48

کوئری بنویسید آیدی، اسم و قیمت تمام محصولاتی که تاکنون سفارش داده نشده‌اند را نمایش دهد.

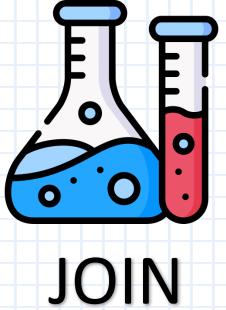




کوئری بنویسید تمام مشخصات محصولاتی که در روزهای ۱۵ تا ۱۸ ماه پنجم سفارش داده شده را نمایش دهد.

49

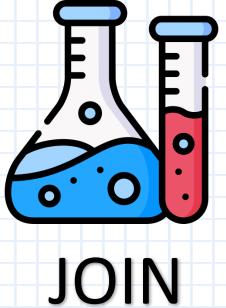




50

کوئری بنویسید تمام مشخصات مشتریانی که در روزهای ۱۵ تا ۱۸ ماه پنجم سفارش داده‌اند را نمایش دهد.

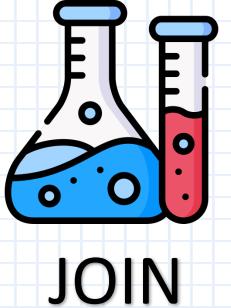




51

کوئری بنویسید اسم و آدرس مشتریانی که محصولی با
قیمت ۴ دلار سفارش داده‌اند را نمایش دهد.

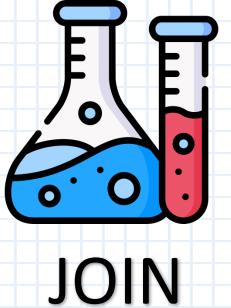




52

کوئی تمام درآمدی که توسط هر مشتری کسب شده
است را نمایش دهد.

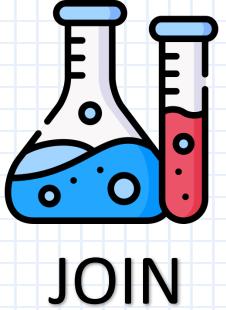




چه محصولی بیشترین میزان فروش را داشته و توسط چه مشتریانی خریداری شده است؟

53

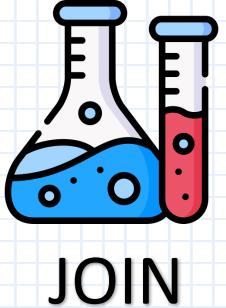




54

کوئری بنویسید که اسم و آیدی مشتریانی که بیش از ۴۰۰ تا سفارش داشته اند را به همراه تعداد سفارش نمایش دهد.





کوئری بنویسید که اسامی مشتریانی که از یک ایالت
هستند را نمایش دهد.

55

```
SELECT A.name as cn1, B.name as cn2, A.state
FROM Customer A, Customer B
WHERE A.Customer_ID <> B.Customer_ID
AND A.state = B.state
```

SUBQUERY



OPERATORS

Operator	Description
ALL	TRUE if all of the subquery values meet the condition
AND	TRUE if all the conditions separated by AND is TRUE
ANY	TRUE if any of the subquery values meet the condition
BETWEEN	TRUE if the operand is within the range of comparisons
EXISTS	TRUE if the subquery returns one or more records
IN	TRUE if the operand is equal to one of a list of expressions
LIKE	TRUE if the operand matches a pattern
NOT	Displays a record if the condition(s) is NOT TRUE
OR	TRUE if any of the conditions separated by OR is TRUE
SOME	TRUE if any of the subquery values meet the condition

ANY ALL

The **ANY** and **ALL** operators allow you to perform a comparison between a single column value and a range of other values.



The **ANY** operator:

- returns a boolean value as a result
- returns TRUE if ANY of the subquery values meet the condition

ANY means that the condition will be true if the operation is true for any of the values in the range.

ALL

The **ALL** operator:

- returns a boolean value as a result
- returns TRUE if ALL of the subquery values meet the condition
- is used with SELECT, WHERE and HAVING statements

ALL means that the condition will be true only if the operation is true for all values in the range.

UNION



Comments



CREATE DB



DROP DB



CREATE TABLE



DROP TABLE



ALTER TABLE

INDEX



SQL INJECTION