מיני פרויקט בבסיסי נתונים

פרויקט: בסיס נתונים עבור בנק

היחידה הנבחרת: מידע ומשכורות של עובדים

מגישים:
אלי ארזי 336301304
אלעד רדומסקי 318645850

תוכן עניינים:

תיאור המערכת:

בפרויקט הזה אנו בונים בסיס נתונים עבור אגף השכר וכוח אדם בבנק.
לבנק ישנם מספר סניפים ולכל סניף עובדים, העובדים מחולקים למנהלים ופקידים ולכן יש יחס של ירושה, לכל סניף יש 10 מחלקות פנימיות.
המערכת פועלת כך:
ישנו עובד, לעובד יש שתי ישויות שיורשות ממנו, עובד יכול להיות מנהל או פקיד, לכל עובד יש משכורת, כמו כן כל עובד עובד בסניף ולכל סניף יש מחלקות (הלוואות, ייעוץ השקעות...).

דוגמא להמחשה:
עובד מסוג פקיד: יעקב
עובד מסוג פקיד: יצחק
עובד מסוג מנהל: אברהם
עובד מסוג מנהל : משה
עובד מסוג מנהל: אהרון

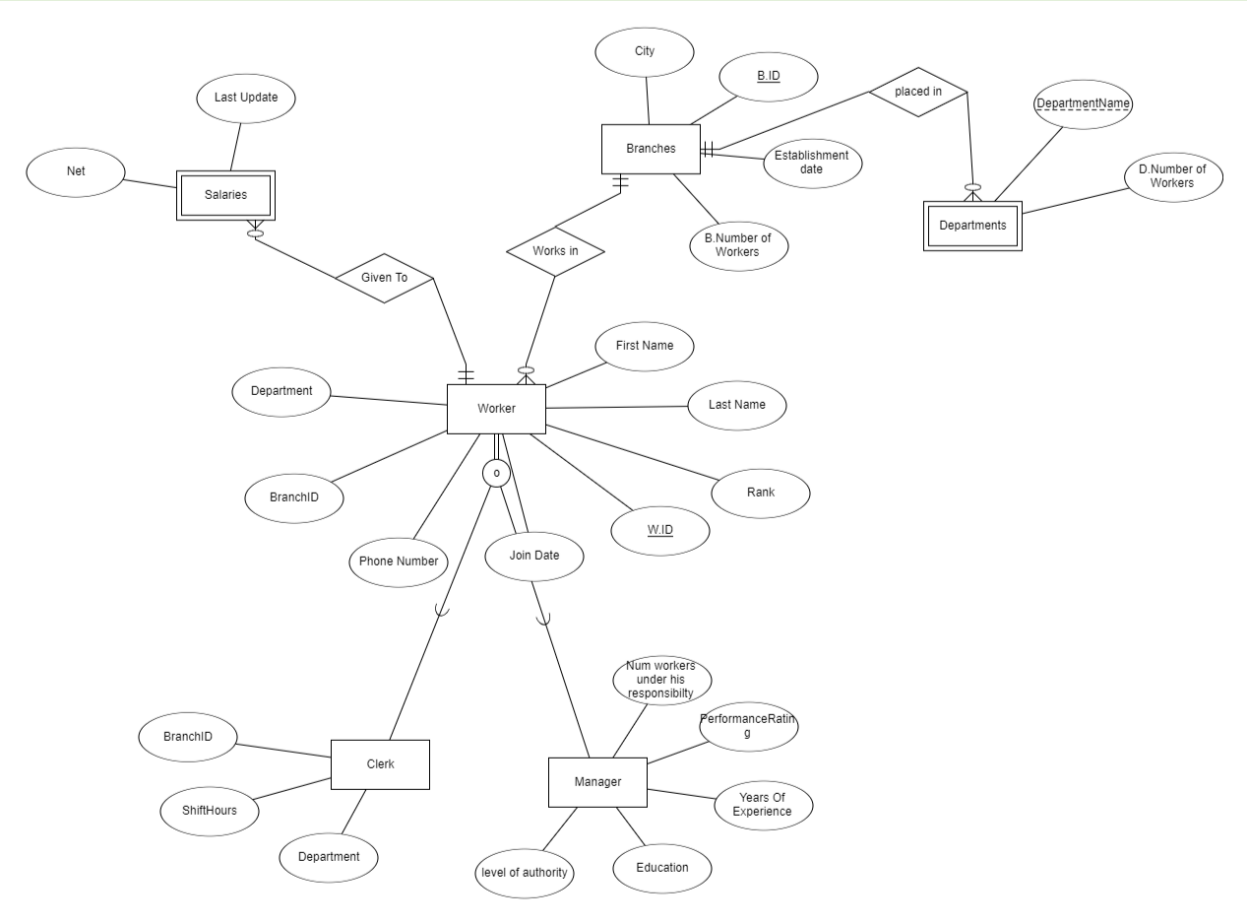
שכר של יעקב 8k
שכר של יצחק 8k
שכר של אברהם 15k
שכר של משה: 15k
שכר של אהרון: 13k

יעקב עובד בסניף: ירוחם כפקיד במחלקת ייעוץ השקעות
משה עובד בסניף: ירוחם כמנהל הסניף
יצחק עובד בסניף: תל אביב כפקיד במחלקת הלוואות
אברהם עובד בסניף: תל אביב, כמנהל הסניף
אהרון עובד בסניף: תל אביב, כמנהל מחלקת הלוואות

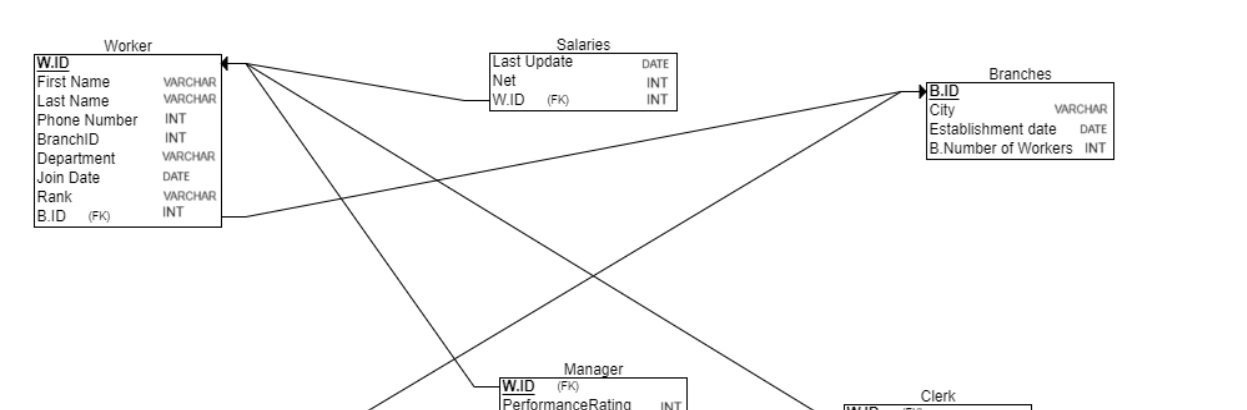
מנהל סניף תל אביב: אברהם
מנהל סניף ירוחם: משה

במילים אחרות ישנם שתי סניפים תל אביב וירוחם, מנהל סניף תל אביב הוא אברהם ומנהל סניף ירוחם הוא משה, לסניף תל אביב במחלקת הלוואות אהרון הוא המנהל, ויצחק עובד כפקיד במחלקה זו, בסניף ירוחם יעקב עובד כפקיד במחלקת ייעוץ השקעות.
המערכת שומרת על מאגר העובדים והשכר שלהם.

ERD


City
B.ID
placed in
DepartmentName
Last Update
Branches
Establishment date
Net
Salaries
D.Number of Workers
Departments
B.Number of Workers
Works in
Given To
First Name
Department
Worker
Last Name
o
BranchID
Rank
W.ID
Phone Number
Join Date
Num workers under his responsibilty
PerformanceRating
BranchID
Clerk
Manager
ShiftHours
Years Of Experience
Department
level of authority
Education

DSD


Worker
W.ID
First Name VARCHAR
Last Name VARCHAR
Phone Number INT
BranchID INT
Department VARCHAR
Join Date DATE
Rank VARCHAR
B.ID (FK) INT
Salaries
Last Update DATE
Net INT
W.ID (FK) INT
Branches
B.ID
City VARCHAR
Establishment date DATE
B.Number of Workers INT
Manager
W.ID (FK)
PerformanceRating INT
Clerk

Create Tables

```
CREATE TABLE `branches` (
 `BranchID` int(11) NOT NULL AUTO_INCREMENT,
 `City` varchar(20) NOT NULL,
 `EstablishmentDate` date NOT NULL,
 `NumberOfWorkers` int(11) NOT NULL,
 PRIMARY KEY (`BranchID`)
)


CREATE TABLE `clerk` (
 `WorkerID` int(11) NOT NULL,
 `Department` varchar(20) NOT NULL,
 `BranchID` int(11) NOT NULL,
 `ShiftHours` int(11) NOT NULL,
 KEY `WorkerID` (`WorkerID`),
 KEY `BranchID` (`BranchID`),
 KEY `Department` (`Department`),
 CONSTRAINT `clerk_ibfk_1` FOREIGN KEY (`WorkerID`) REFERENCES `worker` (`WorkerID`),
 CONSTRAINT `clerk_ibfk_2` FOREIGN KEY (`BranchID`) REFERENCES `branches` (`BranchID`),
 CONSTRAINT `clerk_ibfk_3` FOREIGN KEY (`Department`) REFERENCES `departments` (`DepartmentName`)
)


CREATE TABLE `departments` (
 `DepartmentName` varchar(20) NOT NULL,
 `BranchID` int(11) NOT NULL,
 `NumberOfWorkers` int(11) NOT NULL,
 PRIMARY KEY (`DepartmentName`,`BranchID`),
 KEY `BranchID` (`BranchID`),
 CONSTRAINT `departments_ibfk_1` FOREIGN KEY (`BranchID`) REFERENCES `branches` (`BranchID`) ON UPDATE
CASCADE
)


CREATE TABLE `manager` (
 `WorkerID` int(11) NOT NULL,
 `NumWorkersUHR` int(11) NOT NULL,
 `PreformanceRating` int(11) NOT NULL,
 `YearsOfExperience` int(11) NOT NULL,
```

```
 `Education` varchar(50) NOT NULL,
 `LevelOfAuthority` varchar(50) NOT NULL,
 PRIMARY KEY (`WorkerID`),
 CONSTRAINT `manager_ibfk_1` FOREIGN KEY (`WorkerID`) REFERENCES `worker` (`WorkerID`)
)


CREATE TABLE `salaries` (
 `WorkerID` int(11) NOT NULL,
 `LastUpdate` date NOT NULL,
 `Net` int(11) NOT NULL,
 PRIMARY KEY (`WorkerID`),
 CONSTRAINT `salaries_ibfk_1` FOREIGN KEY (`WorkerID`) REFERENCES `worker` (`WorkerID`) ON UPDATE NO
ACTION
)

CREATE TABLE `worker` (
 `WorkerID` int(11) NOT NULL AUTO_INCREMENT,
 `FirstName` varchar(20) NOT NULL,
 `LastName` varchar(20) NOT NULL,
 `JoinDate` date NOT NULL,
 `PhoneNumber` int(11) NOT NULL,
 `BranchID` int(11) NOT NULL,
 `Department` varchar(20) NOT NULL,
 `Rank` varchar(20) NOT NULL DEFAULT 'Junior',
 PRIMARY KEY (`WorkerID`)
)
```

Python script-data generator

Salaries, Manager and Clerk Tables:

```python
import random
from datetime import datetime, timedelta
# Function to generate random dates
def random_date(start, end):
    return (start + timedelta(days=random.randint(0, (end - start).days))).strftime('%Y-%m-%d')
# Function to calculate Net salary from Gross salary
def calculate_net_salary(gross_salary):
    # Example formula: net salary is gross salary minus 20% taxes
    net_salary = int(gross_salary * 0.80)
    return net_salary
department_names = ['Finance', 'HR', 'IT', 'CustomerService', 'Operations', 'Marketing', 'Legal',
'Compliance', 'Audit', 'Risk']
# Example educations and authority levels
educations = ["Bachelor's Degree", "Master's Degree", "PhD", "Associate Degree", "High School Diploma"]
authority_levels = [
    'Manage IT Department', 'Manage HR Department', 'Manage Finance Department',
    'Manage Operations', 'Manage Customer Service', 'Manage Marketing'
]
# Date range for last update
start_date = datetime.strptime('01-01-1990', '%d-%m-%Y')
end_date = datetime.strptime('26-05-2024', '%d-%m-%Y')
# Salaries Table
salary_insert_commands = []
for worker_id in range(1, 43571):
    gross_salary = random.randint(30000, 120000)  # Random gross salary between 30k and 120k
    net_salary = calculate_net_salary(gross_salary)
    last_update = random_date(start_date, end_date)
    salary_insert_commands.append(f"INSERT INTO Salaries (WorkerID, Gross, Net, LastUpdate) VALUES
({worker_id}, {gross_salary}, {net_salary}, '{last_update}');")
# Manager Table
manager_insert_commands = []
num_managers = int(0.1 * 43570)  # Assuming 10% of workers are managers
manager_worker_ids = random.sample(range(1, 43571), num_managers)  # Unique WorkerIDs for managers
for worker_id in manager_worker_ids:
    num_workers_uhr = random.randint(5, 50)  # Number of workers under responsibility
    performance_rating = random.randint(1, 100)
    years_of_experience = random.randint(1, 40)
    education = random.choice(educations).replace("'", "''")
    level_of_authority = random.choice(authority_levels).replace("'", "''")
    manager_insert_commands.append(f"INSERT INTO Manager (WorkerID, NumWorkersUHR, PreformanceRating,
YearsOfExperience, Education, LevelOfAuthority) VALUES ({worker_id}, {num_workers_uhr},
{performance_rating}, {years_of_experience}, '{education}', '{level_of_authority}');")
# Clerk Table
clerk_insert_commands = []
all_worker_ids = set(range(1, 43571))
clerk_worker_ids = all_worker_ids - set(manager_worker_ids)  # WorkerIDs for clerks, excluding managers
for worker_id in clerk_worker_ids:
```

Branches, Departments and Workers Tables:

```python
import random
from datetime import datetime, timedelta

cities = ['jerusalem', 'TelAviv', 'ramle', 'Haifa', 'BeerSheva', 'Netanya', 'Petahtikva', 'RamatGan',
'Yavne', 'Ashdod', 'Ashkelon', 'Hadera', 'KiryatShmona', 'Afula', 'lod', 'Herzelia', 'KfarSaba',
'Raanana', 'Teveriya', 'MaaleEdomim']
department_names = ['Finance', 'HR', 'IT', 'CustomerService', 'Operations', 'Marketing', 'Legal',
'Compliance', 'Audit', 'Risk']

first_names = ['James', 'John', 'Robert', 'Michael', 'William', 'David', 'Richard', 'Joseph',
'Charles', 'Thomas','Christopher', 'Daniel', 'Matthew', 'Anthony', 'Mark', 'Donald', 'Steven', 'Paul',
'Andrew', 'Joshua','Kenneth', 'Kevin', 'Brian', 'George', 'Edward', 'Ronald', 'Timothy', 'Jason',
'Jeffrey', 'Ryan','Jacob', 'Gary', 'Nicholas', 'Eric', 'Jonathan', 'Stephen', 'Larry', 'Justin',
'Scott', 'Brandon','Benjamin', 'Samuel', 'Gregory', 'Frank', 'Alexander', 'Raymond', 'Patrick', 'Jack',
'Dennis', 'Jerry','Tyler', 'Aaron', 'Jose', 'Adam', 'Nathan', 'Henry', 'Douglas', 'Zachary', 'Peter',
'Kyle','Walter', 'Ethan', 'Jeremy', 'Harold', 'Keith', 'Christian', 'Roger', 'Noah', 'Gerald',
'Carl','Terry', 'Sean', 'Austin', 'Arthur', 'Lawrence', 'Jesse', 'Dylan', 'Bryan', 'Joe',
'Jordan','Billy', 'Bruce', 'Albert', 'Willie', 'Gabriel', 'Logan', 'Alan', 'Juan', 'Wayne',
'Roy','Ralph', 'Randy', 'Eugene', 'Carlos', 'Russell', 'Bobby', 'Victor', 'Martin', 'Ernest',
'Phillip']

last_names = ['Smith', 'Johnson', 'Williams', 'Brown', 'Jones', 'Garcia', 'Miller', 'Davis',
'Rodriguez', 'Martinez','Hernandez', 'Lopez', 'Gonzalez', 'Wilson', 'Anderson', 'Thomas', 'Taylor',
'Moore', 'Jackson', 'Martin','Lee', 'Perez', 'Thompson', 'White', 'Harris', 'Sanchez', 'Clark',
'Ramirez', 'Lewis', 'Robinson','Walker', 'Young', 'Allen', 'King', 'Wright', 'Scott', 'Torres',
'Nguyen', 'Hill', 'Flores','Green', 'Adams', 'Nelson', 'Baker', 'Hall', 'Rivera', 'Campbell',
'Mitchell', 'Carter', 'Roberts','Gomez', 'Phillips', 'Evans', 'Turner', 'Diaz', 'Parker', 'Cruz',
'Edwards', 'Collins', 'Reyes','Stewart', 'Morris', 'Morales', 'Murphy', 'Cook', 'Rogers', 'Gutierrez',
'Ortiz', 'Morgan', 'Cooper','Peterson', 'Bailey', 'Reed', 'Kelly', 'Howard', 'Ramos', 'Kim', 'Cox',
'Ward', 'Richardson','Watson', 'Brooks', 'Chavez', 'Wood', 'James', 'Bennett', 'Gray', 'Mendoza',
'Ruiz', 'Hughes','Price', 'Alvarez', 'Castillo', 'Sanders', 'Patel', 'Myers', 'Long', 'Ross', 'Foster',
'Jimenez']

def random_date(start, end):
    return (start + timedelta(days=random.randint(0, (end - start).days))).strftime('%Y-%m-%d')

def random_phone_number():
    return f"05{random.randint(0, 9)}-{random.randint(1000000, 9999999)}"

# Branches Table
branch_insert_commands = []
branch_data = []
start_date = datetime.strptime('01-01-1990', '%d-%m-%Y')
end_date = datetime.strptime('26-05-2024', '%d-%m-%Y')

for branch_id in range(1, 401):
    city = random.choice(cities)
    establishment_date = random_date(start_date, end_date)
    number_of_workers = random.randint(20, 200)
    branch_insert_commands.append(f"INSERT INTO Branches (city, EstablishmentDate, NumberOfWorkers)
VALUES ('{city}', '{establishment_date}', {number_of_workers});")
    branch_data.append((branch_id, number_of_workers))

# Departments Table
department_insert_commands = []
department_data = []
for branch_id, num_workers in branch_data:
    department_workers = [num_workers // 10 + (1 if x < num_workers % 10 else 0) for x in range(10)]
    for dept_index, dept_name in enumerate(department_names):
        department_insert_commands.append(f"INSERT INTO Departments (DepartmentName, BranchID,
NumberOfWorkers) VALUES ('{dept_name}', {branch_id}, {department_workers[dept_index]});")
        department_data.append((dept_name, branch_id, department_workers[dept_index]))

# Workers Table
worker_insert_commands = []
worker_id = 1
for dept_name, branch_id, dept_workers in department_data:
    for _ in range(dept_workers):
        first_name = random.choice(first_names)
        last_name = random.choice(last_names)
        join_date = random_date(start_date, end_date)
        phone_number = random_phone_number()
        worker_insert_commands.append(f"INSERT INTO Worker (FirstName, LastName, JoinDate, PhoneNumber,
BranchID, Department) VALUES ('{first_name}', '{last_name}', '{join_date}', '{phone_number}'
```

DESC commands:

```
DESC branches;
```

Field	Type	Null	Key	Default	Extra
BranchID	int(11)	NO	PRI	NULL	auto_increment
City	varchar(20)	NO		NULL	
EstablishmentDate	date	NO		NULL	
NumberOfWorkers	int(11)	NO		NULL	

```
DESC departments;
```

Field	Type	Null	Key	Default	Extra
DepartmentName	varchar(20)	NO	PRI	NULL	
BranchID	int(11)	NO	PRI	NULL	
NumberOfWorkers	int(11)	NO		NULL	

```
DESC worker;
```

Field	Type	Null	Key	Default	Extra
WorkerID	int(11)	NO	PRI	NULL	auto_increment
FirstName	varchar(20)	NO		NULL	
LastName	varchar(20)	NO		NULL	
JoinDate	date	NO		NULL	
PhoneNumber	int(11)	NO		NULL	
BranchID	int(11)	NO		NULL	
Department	varchar(20)	NO		NULL	
Rank	varchar(20)	NO		Junior	

```
DESC salaries;
```

Field	Type	Null	Key	Default	Extra
WorkerID	int(11)	NO	PRI	NULL	
LastUpdate	date	NO		NULL	
Net	int(11)	NO		NULL	

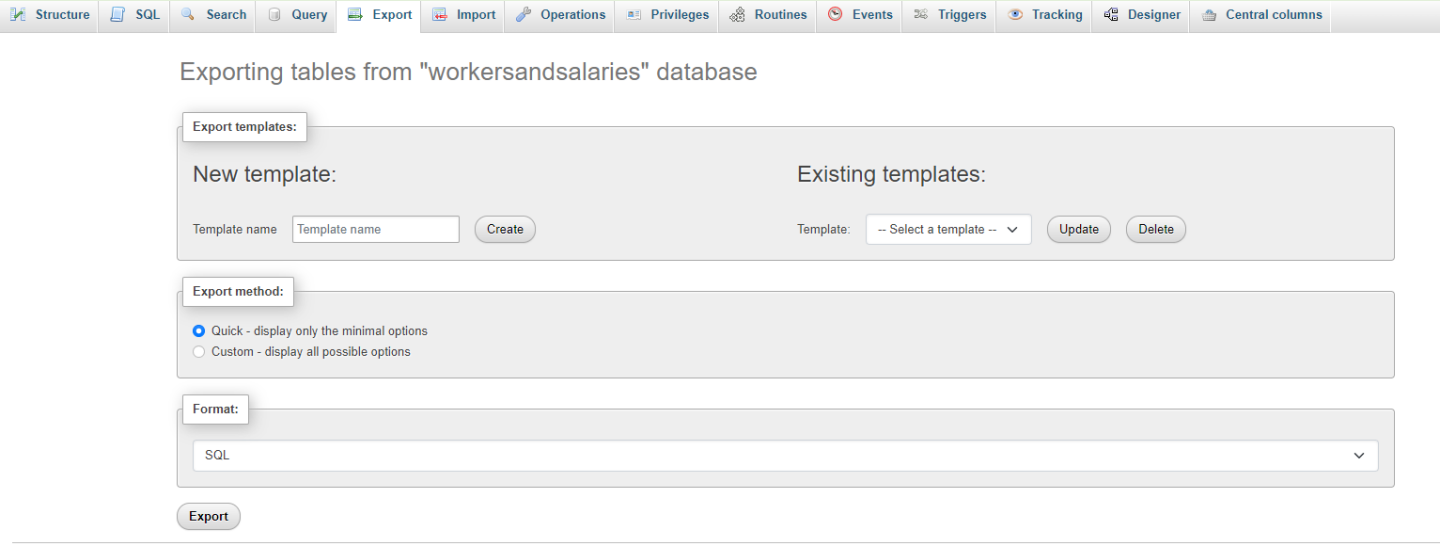
```
DESC manager;
```

Field	Type	Null	Key	Default	Extra
WorkerID	int(11)	NO	PRI	NULL	
NumWorkersUHR	int(11)	NO		NULL	
PreformanceRating	int(11)	NO		NULL	
YearsOfExperience	int(11)	NO		NULL	
Education	varchar(50)	NO		NULL	
LevelOfAuthority	varchar(50)	NO		NULL	

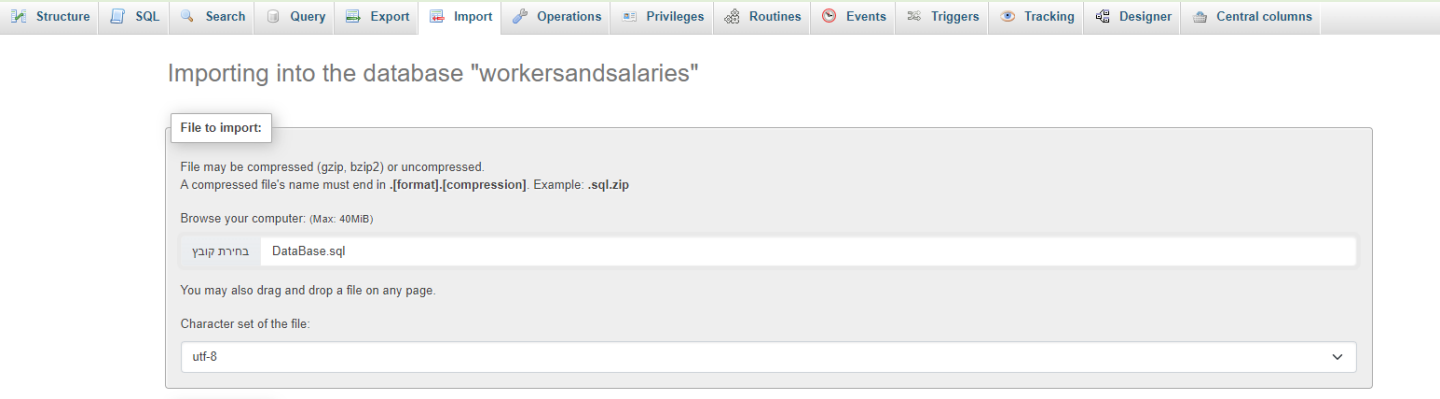
```
DESC clerk;
```

Field	Type	Null	Key	Default	Extra
WorkerID	int(11)	NO	MUL	NULL	
Department	varchar(20)	NO	MUL	NULL	
BranchID	int(11)	NO	MUL	NULL	
ShiftHours	int(11)	NO		NULL	

גיבוי בסיס הנתונים:



שחזור בסיס הנתונים:



שלב 2

שאילתת select 1:

שאילתה שמחזירה את המנהל עם הנסיון והדירוג הכי גבוהים ופולטת מידע על הסניף שלו.

```
SELECT m.WorkerID, w.FirstName, w.LastName, w.Rank, b.BranchID, b.City, b.EstablishmentDate
FROM manager m
JOIN worker w ON m.WorkerID = w.WorkerID
JOIN branches b ON w.BranchID = b.BranchID
WHERE w.Rank = 'Director'
ORDER BY m.YearsOfExperience DESC, m.PreformanceRating DESC
LIMIT 1;
```

השאילתה מצרפת את הטבלה manager עם worker עי התאמת עמודת workerID, ואז מצרפת את worker עם branches בעזרת branchID, ואז מסננת לפי דרגה (rank), ומסדרת את שנות הניסיון בסדר יורד, במקרה של שוויון השאילתה מסדרת לפי דירוג (performance rating).
כלומר השאילתה הזאת מחפשת את פרטי המנהל עם מס הכי גבוה של שנות נסיון עם הדירוג הגבוה ביותר ועם הדרגה הכי גבוהה (director) השאילתה מחזירה את הפרטים של אותו מנהל כולל מידע על המחלקה בו הוא עובד.

הרצה על בסיס הנתונים:

Showing rows 0 - 0 (1 total, Query took 0.0020 seconds.)

```
SELECT m.WorkerID, w.FirstName, w.LastName, w.Rank, b.BranchID, b.City, b.EstablishmentDate
FROM manager m
JOIN worker w ON m.WorkerID = w.WorkerID
JOIN branches b ON w.BranchID = b.BranchID
WHERE w.Rank = 'Director'
ORDER BY m.YearsOfExperience DESC, m.PreformanceRating DESC
LIMIT 1;
```

[Edit inline] [Edit] [Create PHP code]

Extra options

WorkerID	FirstName	LastName	Rank	BranchID	City	EstablishmentDate
21878	Gary	Martinez	Director	206	jerusalem	2005-04-08

Query results operations

שאילתת select 2:

שאילתה המחזירה את מספר העובדים הכולל בכל מחלקה ואת העיר בו היא ממוקמת - עבור סניפים שהוקמו לאחר 2010.

```
SELECT d.DepartmentName, b.City, COUNT(w.WorkerID) AS TotalWorkers
  FROM departments d
  JOIN branches b ON d.BranchID = b.BranchID
  JOIN worker w ON d.DepartmentName = w.Department AND d.BranchID = w.BranchID
  WHERE YEAR(b.EstablishmentDate) > 2010
  GROUP BY d.DepartmentName, b.City
  ORDER BY TotalWorkers DESC;
```

השאילתה לוקחת שם מחלקה, עיר, וסופרת כמה עובדים יש בכל מחלקה וקוראת לסכום TotalWorkers, השאילתה מצרפת את טבלת הסניפים עם טבלת המחלקות על בסיס departmentName ו branchID, ואז מצרפת את טבלת המחלקות עם טבלת העובדים על בסיס departmentName ו branchID, לאחר מכן יש פילטר שלוקח רק את הסניפים שנוסדו לאחר 2010, מקבצת את שם הסניף לעיר ומסדרת בסדר יורד על בסיס מס העובדים.

הרצה על בסיס הנתונים:

Showing rows 0 - 24 (200 total, Query took 0.0524 seconds.) [TotalWorkers: **173... - 119...**]

```
SELECT d.DepartmentName, b.City, COUNT(w.WorkerID) AS TotalWorkers
  FROM departments d
  JOIN branches b ON d.BranchID = b.BranchID
  JOIN worker w ON d.DepartmentName = w.Department AND d.BranchID = w.BranchID
  WHERE YEAR(b.EstablishmentDate) > 2010
  GROUP BY d.DepartmentName, b.City
  ORDER BY TotalWorkers DESC;
```

[Edit inline] [Edit] [Create PHP code]

1 > >> | Show all | Number of rows: 25 Filter rows: Search this table

Extra options

DepartmentName	City	TotalWorkers 1
Finance	RamatGan	173
HR	RamatGan	172
CustomerService	RamatGan	170
IT	RamatGan	170
Operations	RamatGan	168

שאילתת select 3:

שאילתה שמוצאת את השכר הממוצע לכל מחלקה, עבור מחלקות עם יותר מ5 עובדים.

```
SELECT w.Department, AVG(s.Net) AS AverageSalary
 FROM worker w
 JOIN salaries s ON w.WorkerID = s.WorkerID
 GROUP BY w.Department
 HAVING COUNT(w.WorkerID) > 5
 ORDER BY AverageSalary DESC;
```

השאילתה לוקחת את עמודת המחלקות מטבלת עובד, ומחשבת את הממוצע מטבלת משכורת וקוראת לזה average salary, לאחר מכן יש איחוד בין טבלת עובד לטבלת משכורת על בסיס workerID, מקבצים על בסיס מחלקה, ישנו פילטר שלוקח רק את הקבוצות עם יותר מ5 עובדים לבסוף מסדרים הכל בסדר יורד על בסיס המשכורת הממוצעת.

הרצה על בסיס הנתונים:

Showing rows 0 - 9 (10 total, Query took 0.0545 seconds.) [AverageSalary: **24818.1234... - 24331.5631...**]

```
SELECT w.Department, AVG(s.Net) AS AverageSalary
 FROM worker w
 JOIN salaries s ON w.WorkerID = s.WorkerID
 GROUP BY w.Department
 HAVING COUNT(w.WorkerID) > 5
 ORDER BY AverageSalary DESC;
```

[Edit inline] [Edit] [Create PHP code]

Show all | Number of rows: 25 Filter rows: Search this table

Extra options

Department	AverageSalary 1
Marketing	24818.1234
Risk	24713.4061
Compliance	24659.1078
CustomerService	24651.7253
Legal	24627.1158

שאילתת select 4:

שאילתה שמחפשת ומחזירה את הפרטים של כל הפקידים בדרגת Senior שעובדים באשקלון ויש להם משמרות של יותר מ30 שעות בשבוע.

```
SELECT c.WorkerID, w.FirstName, w.LastName, w.Rank, c.ShiftHoursPW,b.BranchID, b.City
 FROM clerk c
 JOIN worker w ON c.WorkerID = w.WorkerID
 JOIN branches b ON c.BranchID = b.BranchID
 WHERE b.city = 'Ashkelon' AND w.Rank = 'Senior' AND c.ShiftHoursPW > 30;
```

לוקחים את עמודת תז של עובד מטבלת עובד פקיד, לוקחים שם שם משפחה ודרגה של עובד מטבלת עובד, לוקחים את שעות משמרת מטבלת עובד פקיד, את הסניף ואת העיר מטבלת סניפים, לאחר מכן מצרפים את טבלת עובד פקיד עם טבלת עובד על בסיס workerID, ומצרפים את הטבלה של עובד פקיד עם טבלת סניפים על בסיס BranchID, לאחר מכן יש פילטר שלוקח רק את הערכים שהעיר היא אשקלון, שהדרגה היא senior ועם משמרת של יותר מ30 שעות בשבוע.

הרצה על בסיס הנתונים:

Showing rows 0 - 24 (124 total, Query took 0.0052 seconds.)

```
SELECT c.WorkerID, w.FirstName, w.LastName, w.Rank, c.ShiftHoursPW,b.BranchID, b.City
 FROM clerk c
 JOIN worker w ON c.WorkerID = w.WorkerID
 JOIN branches b ON c.BranchID = b.BranchID
 WHERE b.city = 'Ashkelon' AND w.Rank = 'Senior' AND c.ShiftHoursPW > 30;
```

[Edit inline] [Edit] [Create PHP code]

1 > >> | Show all | Number of rows: 25 Filter rows: Search this table

Extra options

WorkerID	FirstName	LastName	Rank	ShiftHoursPW	BranchID	City
14601	Donald	Ross	Senior	36	15	Ashkelon
19302	Tyler	Garcia	Senior	35	15	Ashkelon
28837	Carlos	Johnson	Senior	40	15	Ashkelon
14601	Donald	Ross	Senior	36	15	Ashkelon
19302	Tyler	Garcia	Senior	35	15	Ashkelon

שאילתת delete 1:

שאילתה שמוחקת את כל העובדים שדרגתם Junior במחלקת פיננסים בעיר לוד.

```
DELETE w
    FROM worker w
    JOIN branches b ON w.BranchID = b.BranchID
    WHERE w.Rank = 'Junior' AND w.Department = 'Finance' AND b.City = 'Lod';
```

לפני ביצוע הפקודה:

Showing rows 0 - 24 (29 total, Query took 0.0124 seconds.)

```
SELECT w.*
FROM worker w
JOIN branches b ON w.BranchID = b.BranchID
WHERE w.Rank = 'Junior' AND w.Department = 'Finance' AND b.City = 'Lod';
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

1 > >> | Show all | Number of rows: 25 Filter rows: Search this t

Extra options

WorkerID	FirstName	LastName	JoinDate	PhoneNumber	BranchID	Department	Rank
40	Scott	James	2019-03-16	0569911972	2	Finance	Junior
47	Ralph	Taylor	2008-05-04	0521429732	2	Finance	Junior
48	Ryan	Lee	2009-07-03	0595563656	2	Finance	Junior
5624	Anthony	Robinson	2010-04-13	0573497112	54	Finance	Junior
9682	Tyler	Cooper	2021-05-16	0583987114	91	Finance	Junior

לאחר ביצוע הפקודה: ניתן לראות כי לא קיימים יותר עובדים בדרגת Junior במחלקת פיננסים בסניף לוד.

```
SELECT w.*
FROM worker w
JOIN branches b ON w.BranchID = b.BranchID
WHERE w.Rank = 'Junior' AND w.Department = 'Finance' AND b.City = 'Lod';
```

Update query | **Submit query**

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0214 seconds.)

שאילתת delete 2:

שאילתה שמוחקת את כל המנהלים עם השכלה תיכונית ועם דירוג נמוך מ30.

```
DELETE FROM manager
WHERE Education = 'High School Diploma' AND PreformanceRating < 30;
```

לפני ביצוע הפקודה:

Showing rows 0 - 24 (235 total, Query took 0.0004 seconds.)

```
SELECT * FROM `manager` WHERE `PreformanceRating` < 30 AND `Education` LIKE 'High School Diploma'
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

1 > >> | Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

←T→	WorkerID	NumWorkersUHR	PreformanceRating	YearsOfExperience	Education	LevelOfAuthority
Edit Copy Delete	26	6	9	40	High School Diploma	Manage Operations
Edit Copy Delete	125	41	11	40	High School Diploma	Manage HR Department
Edit Copy Delete	203	44	17	19	High School Diploma	Manage IT Department
Edit Copy Delete	260	16	5	5	High School Diploma	Manage HR Department
Edit Copy Delete	317	42	19	3	High School Diploma	Manage Operations

לאחר ביצוע הפקודה:

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0012 seconds.)

```
SELECT * FROM manager WHERE Education = 'High School Diploma' AND PreformanceRating < 30;
```

שאילתת update 1:

שאילתה שמקדמת את כל העובדים בדרגת junior לדרגת senior בתנאי שהם עובדים בבנק יותר מ5 שנים.

```
UPDATE worker
 SET Rank = 'Senior'
 WHERE Rank = 'Junior' AND DATEDIFF(CURDATE(), JoinDate) > 5 * 365;
```

טבלה לפני ההרצה:

```
SELECT * FROM worker WHERE Rank = 'Junior' AND DATEDIFF(CURDATE(), JoinDate) > 5 * 365;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

1 > >> | Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

	WorkerID	FirstName	LastName	JoinDate	PhoneNumber	BranchID	Department	Rank
Edit Copy Delete	2	Joshua	Nguyen	2000-04-01	0595214287	1	Finance	Junior
Edit Copy Delete	5	Henry	Williams	1996-07-27	0599489637	1	HR	Junior
Edit Copy Delete	17	Carlos	Brooks	2017-08-19	0588836261	1	Operations	Junior
Edit Copy Delete	24	James	Ross	2007-10-30	0563079811	1	Marketing	Junior
Edit Copy Delete	25	Peter	Torres	2015-02-25	0569864288	1	Legal	Junior
Edit Copy Delete	27	Paul	Cox	1996-09-22	0512065484	1	Legal	Junior

לאחר ההרצה:

	WorkerID	FirstName	LastName	JoinDate	PhoneNumber	BranchID	Department	Rank
Edit Copy Delete	2	Joshua	Nguyen	2000-04-01	0595214287	1	Finance	Senior
Edit Copy Delete	5	Henry	Williams	1996-07-27	0599489637	1	HR	Senior
Edit Copy Delete	13	Gabriel	Allen	2000-01-25	0512855479	1	CustomerService	Senior
Edit Copy Delete	14	Arthur	King	2011-06-25	0566004914	1	CustomerService	Senior
Edit Copy Delete	17	Carlos	Brooks	2017-08-19	0588836261	1	Operations	Senior
Edit Copy Delete	24	James	Ross	2007-10-30	0563079811	1	Marketing	Senior
Edit Copy Delete	25	Peter	Torres	2015-02-25	0569864288	1	Legal	Senior
Edit Copy Delete	27	Paul	Cox	1996-09-22	0512065484	1	Legal	Senior

שאילתת update 2:

שאילתה שמעלה שכר לכל העובדים שהם directors בעשר אחוז, בתנאי שהסניפים שלהם בירושלים.

```sql
UPDATE salaries s
JOIN worker w ON s.WorkerID = w.WorkerID
SET s.Net = s.Net * 1.10
WHERE w.Rank = 'Director' AND w.BranchID IN (
    SELECT BranchID
    FROM branches
    WHERE City = 'Jerusalem'
);
```

השאילתה לוקחת את הדרגה מטבלת עובד, ואת מספר הסניף, השאילתה בודקת מה מספרי הסניפים שנמצאים בירושלים, השאילתה מוצאת את השכר של כל עובד מטבלת השכר בעזרת צירוף של טבלת עובד ושכר על בסיס תז, ומעלה את השכר ב 10%.

לפני:

WorkerID	FirstName	Net
5363	Anthony	11013
5372	Ryan	15484
5381	Jason	24787
5384	Zachary	32340
5392	Christian	19539
5394	Ethan	31236
5401	Benjamin	12910

אחרי:

WorkerID	FirstName	Net
5363	Anthony	12114
5372	Ryan	17032
5381	Jason	27266
5384	Zachary	35574
5392	Christian	21493
5394	Ethan	34360
5401	Benjamin	14201

שאילתה 1 עם פרמטר:

השאילתה משתמשת בפרמטר אחד BranchID, היא בוחרת נתונים של עובדים מסניף מסוים, בשאילתה הזו 101.

```
PREPARE stmt FROM '
SELECT w.WorkerID, w.FirstName, w.LastName, w.Department, w.Rank
FROM worker w WHERE w.BranchID = ?';

SET @branch_id = 101;
EXECUTE stmt USING @branch_id;
DEALLOCATE PREPARE stmt;
```

ראשית השאילתה מכינה את ה"שאילתה" ונותנת את זה ל handler בשם stmt, במקום המתאים עבור branchID ישנו סימן שאלה וזה בעצם מחזיק מקום (placeholder).
לאחר מכן אנו נותנים לערך branchesID את הערך שאנחנו רוצים במקרה הזה 101, ואז אומרים לשאילתה להפעיל את עצמה עם הערך של branchID שהגדרנו, השאילתה תופעל כאשר branchID = 101, לבסוף משחררים את stmt.

Showing rows 0 - 24 (192 total, Query took 0.0145 seconds.)

```
EXECUTE stmt USING @branch_id;
```

[Edit inline] [Edit] [Create PHP code]

Extra options

WorkerID	FirstName	LastName	Department	Rank
10775	Lawrence	Walker	Finance	Intern
10776	Charles	Foster	Finance	MidLevel
10777	Carlos	Torres	Finance	Senior
10778	Jonathan	Robinson	Finance	Junior
10779	Billy	Mitchell	Finance	Senior

שאילתה 2 עם פרמטר:

השאילתה מקבלת רשימה בתור פרמטר ומחזירה את מספר העובדים במחלקות הללו ומקבצת את הפלט לפי מחלקה.

```
PREPARE stmt FROM '
SELECT w.Department, COUNT(w.WorkerID) AS TotalWorkers FROM worker w
WHERE w.Department IN (?, ?, ?)
GROUP BY w.Department';

SET @dept1 = 'Finance', @dept2 = 'HR', @dept3 = 'IT';
EXECUTE stmt USING @dept1, @dept2, @dept3;
DEALLOCATE PREPARE stmt;
```

השאילתה ראשית מכינה את ה"שאילתה" והכל מוחזק על ידי stmt, השאילתה הזו לוקחת רשימה של מחלקות וסופרת את סכום העובדים במחלקה, הפלט מקובץ לפי מחלקה, לאחר מכן מגדירים משתנים עם ערכים כל ערך יהיה מחלקה, ואז מפעילים את stmt עם הערכים שהגדרנו, בסוף מוחקים את המחזיק מקום (כדי לשחרר מקום).

הצבנו את הערכים "Finance" , "HR" , "IT" בפרמטרים והרצנו

Department	TotalWorkers
Finance	4543
HR	4505
IT	4454

שאילתה 3 עם פרמטר:

השאילתה מקבלת כפרמטר ערך של תאריך, השאילתה מחזירה פרטים של עובד (תז שם ותאריך הצטרפות) עבור כל העובדים שהצטרפו לאחר התאריך שקיבלה.

```
PREPARE stmt FROM '
SELECT w.WorkerID, w.FirstName, w.LastName, w.JoinDate
FROM worker w
WHERE w.JoinDate > ?';

SET @join_date = '2020-01-01';
EXECUTE stmt USING @join_date;
DEALLOCATE PREPARE stmt;
```

ראשית השאילתה מכינה את ה"שאילתה" שלוקחת מידע של עובד ובמקום של joindate יש סימן של גדול מ?
והמשתנה stmt מחזיק אותה, לאחר מכן מגדירים משתנה שמחזיק תאריך, ואז מפעילים את השאילתה עם המשתנה שהגדרנו, והיא תחזיר את כל העובדים שהצטרפו לארגון לאחר התאריך הנתון.

הצבנו את התאריך 1 לינואר 2020 בפרמטר והרצנו:

WorkerID	FirstName	LastName	JoinDate
1	Randy	Murphy	2022-02-20
8	Justin	Scott	2022-01-19
11	Donald	Kim	2024-02-01
16	Stephen	Howard	2024-01-03
18	Victor	Lewis	2023-01-21

שאילתה 4 עם פרמטר:

השאילתה הזו מקבלת כפרמטר טווח של תאריכים (ת.התחלה ת.סוף) ופולטת תז, תאריך עדכון אחרון, ונטו של משכורת שעודכנו בטווח הנ"ל, כאשר תאריך עדכון אחרון צריך להיות בטווח תאריכים שקיבלנו כפרמטר.

```
PREPARE stmt FROM '
SELECT s.WorkerID, s.LastUpdate, s.Net
FROM salaries s
WHERE s.LastUpdate BETWEEN ? AND ?';

SET @start_date = '2023-01-01', @end_date = '2023-12-31';
EXECUTE stmt USING @start_date, @end_date;
DEALLOCATE PREPARE stmt;
```

השאילתה מכינה את ה"שאילתה" שלוקחת תז תאריך עדכון אחרון ומשכורת מטבלת משכורת, כאשר תאריך עדכון אחרון צריך להיות בין שתי סימני שאלה שהם פרמטרים את כל זה היא שמה במחזיק מקום בשם stmt.
לאחר מכן השאילתה מגדירה שתי תאריכים ונותנת להם ערכי תאריך, ואז stmt מופעל עם שתי הפרמטרים שיוצבו במקום סימני השאלה, והפלט יהיה פרטי עובד שהמשכורת שלו עודכנה בין שתי תאריכים נתונים.

הצבנו את התאריך 1 לינואר 2023 כתאריך התחלה ו 31 לדצמבר 2023 כתאריך סוף בפרמטר והרצנו:

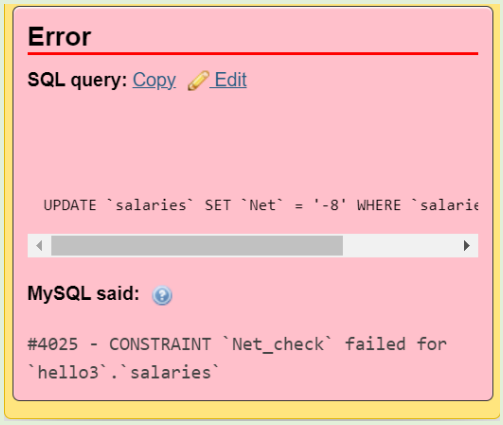
WorkerID	LastUpdate	Net
5	2023-12-29	9849
14	2023-06-16	19763
16	2023-02-15	35119
49	2023-06-02	22438
54	2023-03-15	30652

אילוץ 1:

אילוץ שכל ערכי השכר יהיו גדולים או שווים מ-0.

```
ALTER TABLE salaries
  ADD CONSTRAINT Net_check CHECK (Net >= 0);
```

נסיון של להכניס ערך -8 לשכר נפגש בשגיאה

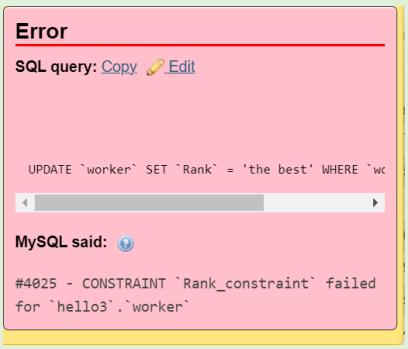


אילוץ 2:

אילוץ שהערך של Rank יהיה ערכים מסויימים בלבד, כלומר אם נכניס עובד בדרגה שהיא לא מבין הדרגות הספציפיות נקבל שגיאה.

```
ALTER TABLE Worker
  ADD CONSTRAINT rank_constraint CHECK (Rank IN ('Junior', 'Intern', 'MidLevel', 'Manager', 'Senior', 'Expert',
'Director'));
```

נסיון של הכנסת ערך דרגה the best נתקל בשגיאה:



אילוץ 3:

מאלץ שלכל סניף יהיה מקסימום של 999 עובדים.

```
ALTER TABLE branches
ADD CONSTRAINT Workers CHECK (NumberOfWorkers <=1000);
```

נסיון להכניס ערך 10000 במספר העובדים בסניף נתקל בשגיאה:

Error

SQL query: Copy Edit

```
UPDATE `branches` SET `NumberOfWorkers` = '10000'
```

MySQL said:

```
#4025 - CONSTRAINT `Workers` failed for
`hello3`.`branches`
```