Python and SQL: intro / SQL platforms

Instructor: Maciej Wilamowski, Ph.D. , mwilamowski@wne.uw.edu.pl

University of Warsaw, Faculty of Economic Sciences

Chair of Microeconomics

Instructor: Robert Wojciechowski, rwojciechowski@wne.uw.edu.pl

University of Warsaw, Faculty of Economic Sciences

Department of Quantitative Finance

Uniwersytet Warszawski
Wydział Nauk
Ekonomicznych

Lab 1

Uniwersytet Warszawski
Wydział Nauk
Ekonomicznych

Course content (part 1):Intro to SQL

1. **Relational model for database management.**
2. **SQL: Table manipulation and basic queries: create/drop table, select, where, insert, update**
3. **SQL: complex queries, joins, stored procedures**
4. **SQL: indexing, triggers**

Course content (part 2):Intro to Python

1. **Preparation of the environment (Ipython Notebook/PyCharm, data structures, debugging).**
2. **Flow control: if, for, while, iterators, error handling. Working with text files.**
3. **Functions and classes.**
4. **Linear algebra with NumPy**
5. **Data handling and wrangling with Pandas.**
6. **Visualization with Seaborn and matplotlib.**
7. **Python in the web: using APIs, JSON, XML, simple web applications.**
8. **Database manipulation with Python.**
9. **Presentations of projects.**

Grading:

- **Final project (60%) : written report + presentation (15 min)**
- **Written test (40%) [practical part +theoretical part]**
- **Activity (up to 10% extra)**
- **The class attendance is mandatory. Four or more unjustified absences signify failure of the course.**

Grade	Total Score %	Description
5	+90%	very good
4+	+80%	better than good
4	+70%	good
3+	+60%	satisfactory
3	+50%	sufficient
2	Less than 51%	fail

Final project requirements:

Goal of the project:

- Prepare own project, which presents how Python is applied to database programming

Possible project topics:

- Python and DBMS in business solutions (Warehouse, Airline reservations system,..)
- Custom Python library (ex. text analyzer) + simple registration website for clients. Website allows clients to enter their personal data and download library.

Each project should :

- be prepared by 1-2 students
- contain codes written in Python + description of functionality and instruction (5 -6 pages)

Deadlines:

- Proposal of the project should be send instructor till the end of November 2018
- Individual defence of the project will be conducted during last labolatories

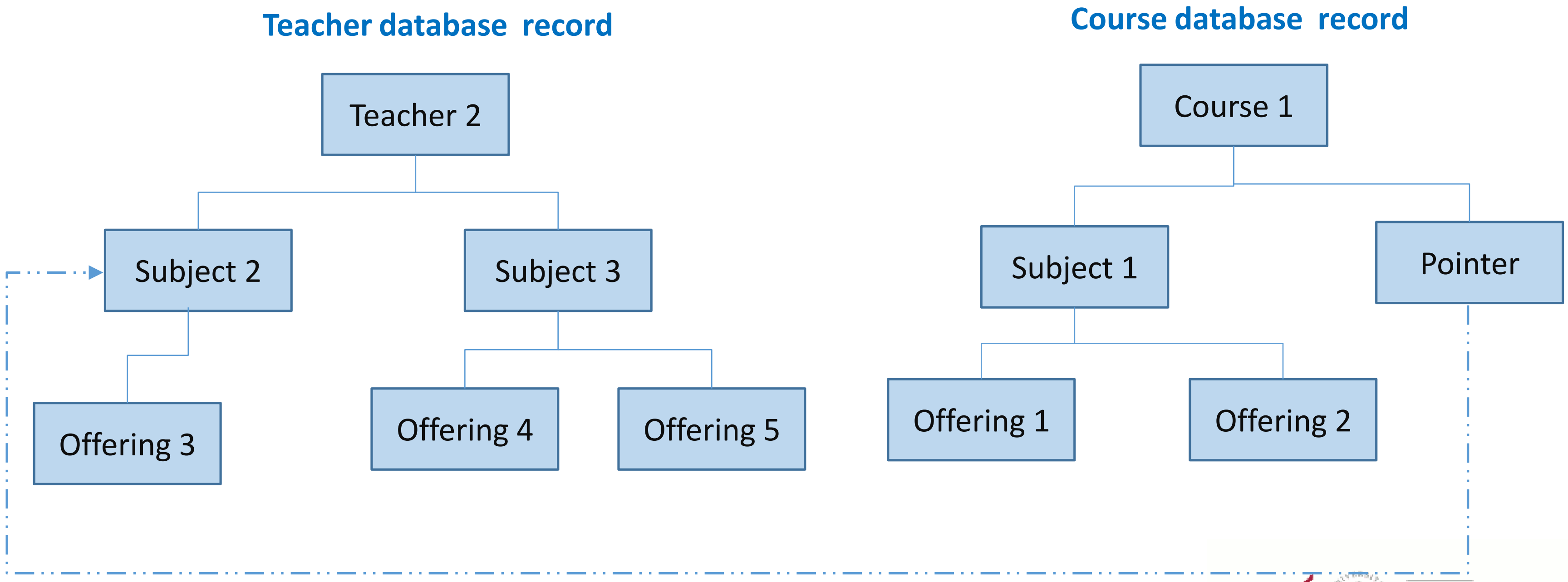
(5-10 min per project)

Database Introduction

- **Database model** :
  - defines the logical design of the data
  - describes the relationships between different part of data
- **most common database models**:
  - ❑Hierarchical Model
  - ❑Network Model
  - ❑Relational Model

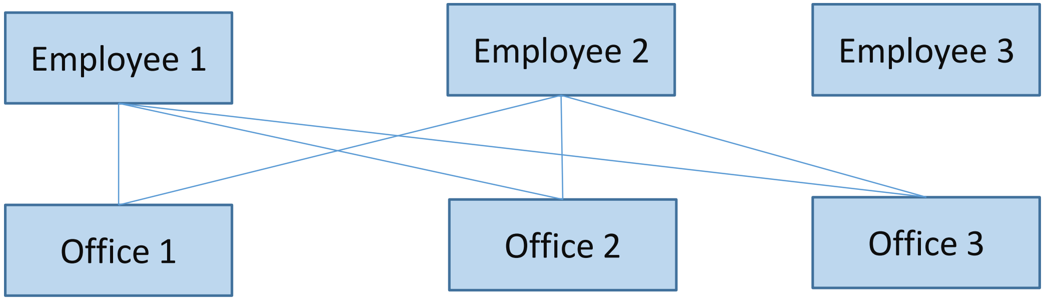
Hierarchical Model

- **The oldest database model**
- **Used mostly in banks**
- **Hierarchical structure is implemented on tree**
- **Each child node has only one parent node**
- **Parent node can have many child nodes**
- **Model efficient but difficult to implement because of its complex structure**



Network Model

- **More flexible than Hierarchical Model**
- **Entities organized in graph**
- **No levels, record can have any number of owners**
- **Number of links between records is high**
- **Databases, which are based on this model are slow**



Relational Model

- First introduced by E.F Codd (1970) in **"A Relational Model of data represented as set of tables"**
- **Table** = collection of data elements organised in terms of rows and columns. Tables allow to represent relations.
- **Record (single row)** = represents set of related data in a table
- **Attribute (column) =** set of value of a particular type in table
- Example: table with 3 records (rows) and 4 attributes (columns)

Id	name	surname	salary
1	Tom	Smith	5000
2	Alex	Crank	6000
3	David	Gold	3000

Database Keys

- **Primary Key –** it is a key that uniquely identify each record in the table. Here {Id}

Id	Name	Surname	Age
0123	Tom	Gold	45
0124	Mark	Brown	21

- **Composite Key -** consist of two or more attributes that uniquely identify an entity occurance. Here {Name,Surname,DateOfBirth}

Name	Surname	DateOfBirth	NrOfCars
Tom	Gold	6.16.1965	2
Mark	Brown	6.16.1965	1
Adam	Gold	23.11.1975	2

Normalization of Database

- **Normalization –** process of organizing the columns and tables of relational database to reduce data redundancy and ensure data integrity
- All four forms of normalization were introduced by Edgar F. Codd
- **Forms of normalization:**
  - ❑1NF (1970)
  - ❑2NF (1971)
  - ❑3NF (1971)
  - ❑BCNF (1974)

Data redundancy:

appears, when the same piece of data is held in two separate places

Data integrity:

overall completeness, accuracy and consistency of data, for example: summed amount of money at two bank accounts should be the same after and before money transfer

First Normal Form (1NF)

- Each row should have a Primary Key that distinguishes it as unique
- Row must not have a column in which more than one value is stored (ex. values:Toyota**,** Fiat in column {Car})
- After the transformation composite Primary Key is {Surname,Car},

before

Surname	Age	Car
Gold	45	Toyota, Fiat
Brown	31	Honda, Renault

after

Surname	Age	Car
Gold	45	Toyota
Gold	45	Fiat
Brown	31	Honda
Brown	31	Renault

Second Normal Form (2NF)

- Must be in 1NF
- 2NF requires that any non-key field be dependent on the entire **Primary Key**
- **Problem:** in the example below, the candidate composite **Primary Key** is {Surname,Car}, but {Age} attribute depends only upon {Surname} attribute
- **Solution**: make single **Primary Key** by dividing input table into two tables, introduce Foreign Key {Surname} in one of the new tables

before (1NF)

Surname	Age	Car
Gold	45	Toyota
Gold	45	Fiat
Brown	31	Honda
Brown	31	Renault

after (2NF)

Surname	Age
Gold	45
Brown	31

ID_Surname _Car	Surname	Car
1	Gold	Toyota
2	Gold	Fiat
3	Brown	Honda
4	Brown	Renault

Third Normal Form (3NF)

Transitive Dependency : exists, when any attribute in a table is dependent upon any other non-key attribute in that table

- Must be in the 2NF (all attributes depend upon Primary Key)
- Transitive dependency must be removed
- In given table, {Student_id} is Primary Key, but {Street}, {City} and {State} depend also upon {ZipCode}. The dependency between {Student_Id} and each of these fileds is called **transitive dependency.**

before (2NF)

Student_Id	Name	Surname	Street	City	State	ZipCode
23	Tom	Gold	Silver Street	Big City	Alabama	23-3472

after (3NF)

Student_Id	Name	Surname	ZipCode
23	Tom	Gold	23-3472

ZipCode	City	State	Street
23-3472	Big City	Alabama	Silver Street

Boyce and Codd Normal Form (BCNF)

- Higher version of the Third Normal form
- Table must be in 3NF form and does not have multiple overlapping candidate keys. For every functional depenedence X->Y,X should be the super key of the table.

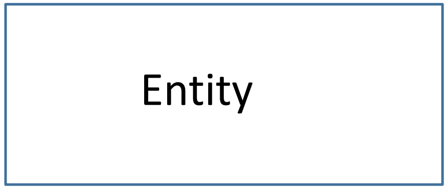
Summary:

[1NF] - data depends on the key
[2NF]- data depends the whole key
[3NF]- data depends on nothing but the key

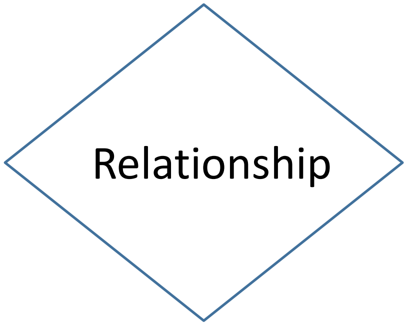
Entity Relationship Diagram (ERD)

- **Entity Relationship Diagram (ERD)** - visual representation of data that describes how data is related to each other

- **Entity** - object or concept about which you want to store information

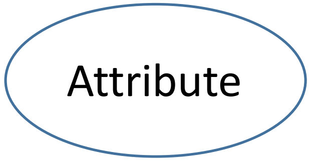


- **Action -** represented by diamond shape, shows how two entities share information in the database

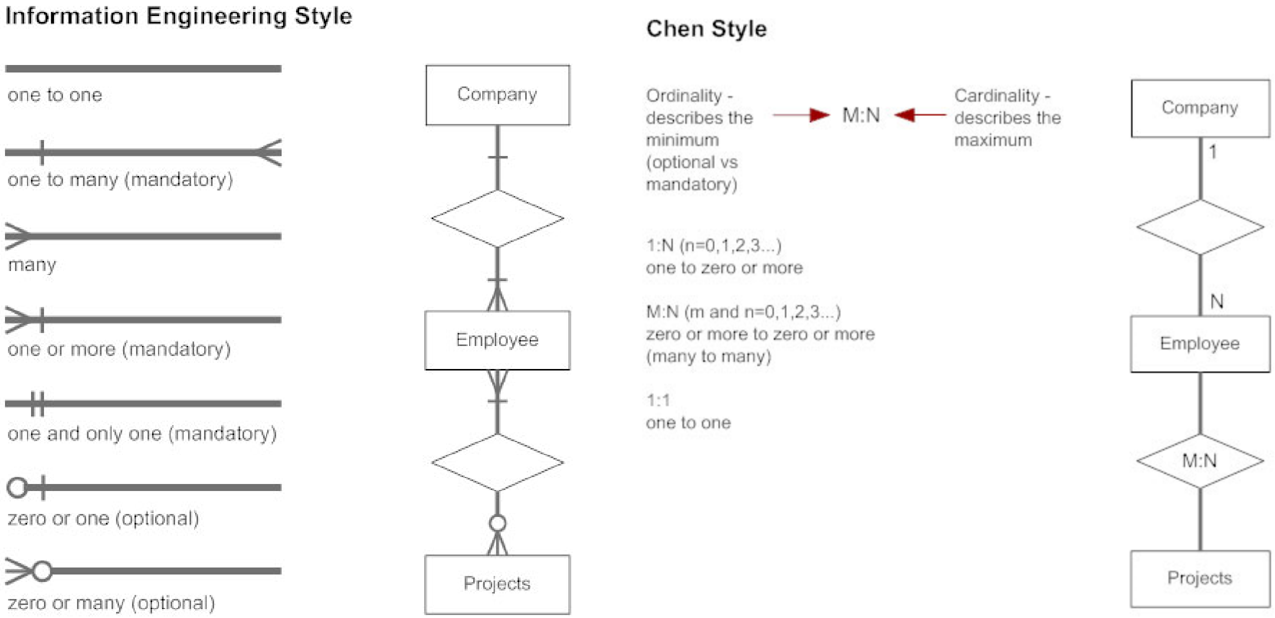


Entity Relationship Diagram

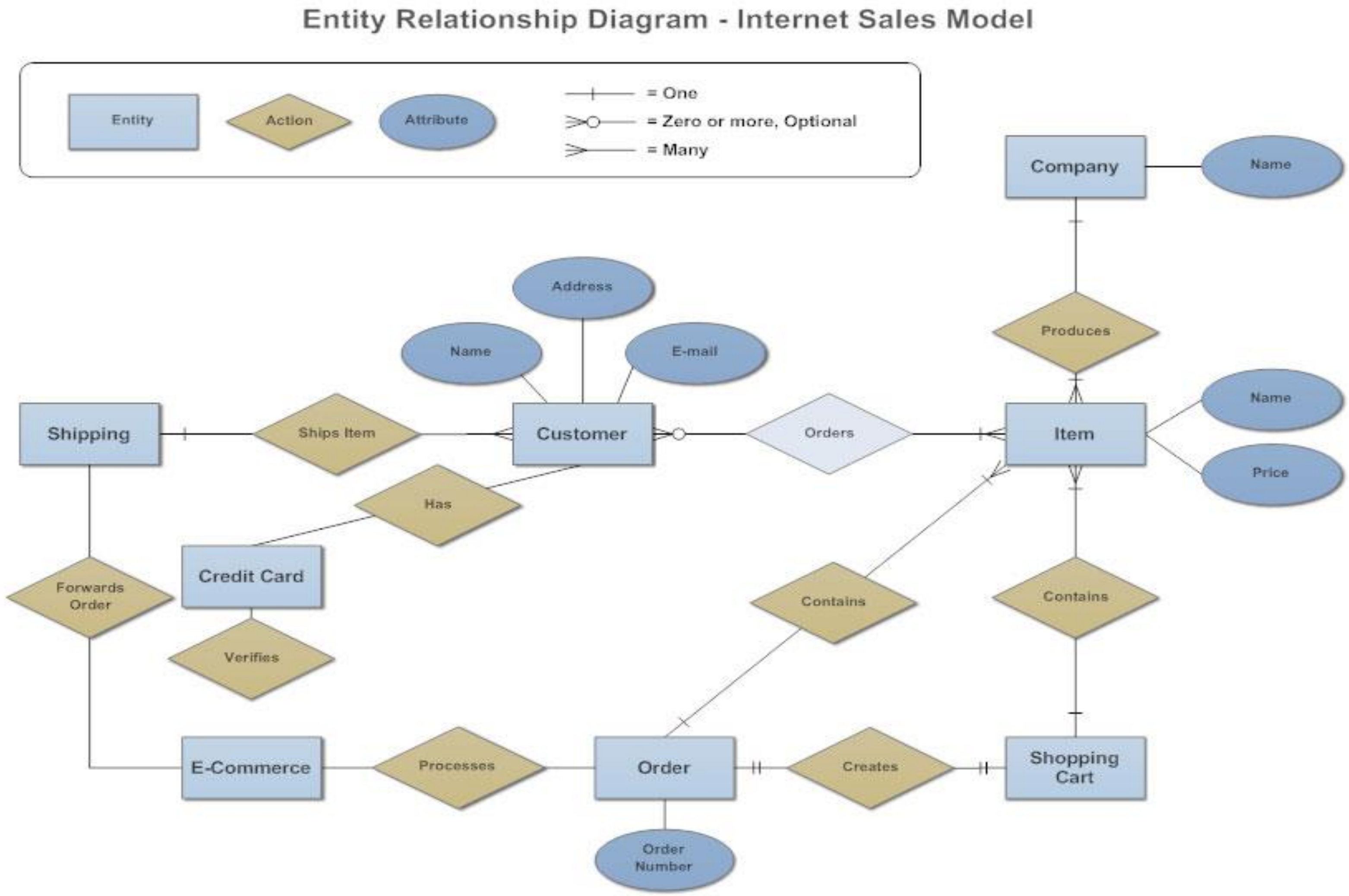
- **Attribute –** represented by oval. Describes single feature of the entity. **A key attribute** is the unique, distinguishing characteristic of the entity, for example: an employee has attribute Pesel number.



- **Relations –** in the context of cardinality, we should specify how many instances of an entity relate to one instance of another entity. There are many notation styles that express cardinality.



ERD Example


Entity Relationship Diagram - Internet Sales Model
Entity
Action
Attribute
= One
= Zero or more, Optional
= Many
Company
Name
Produces
Address
Name
E-mail
Shipping
Ships Item
Customer
Orders
Item
Name
Price
Has
Credit Card
Forwards Order
Verifies
Contains
Contains
E-Commerce
Processes
Order
Creates
Shopping Cart
Order Number

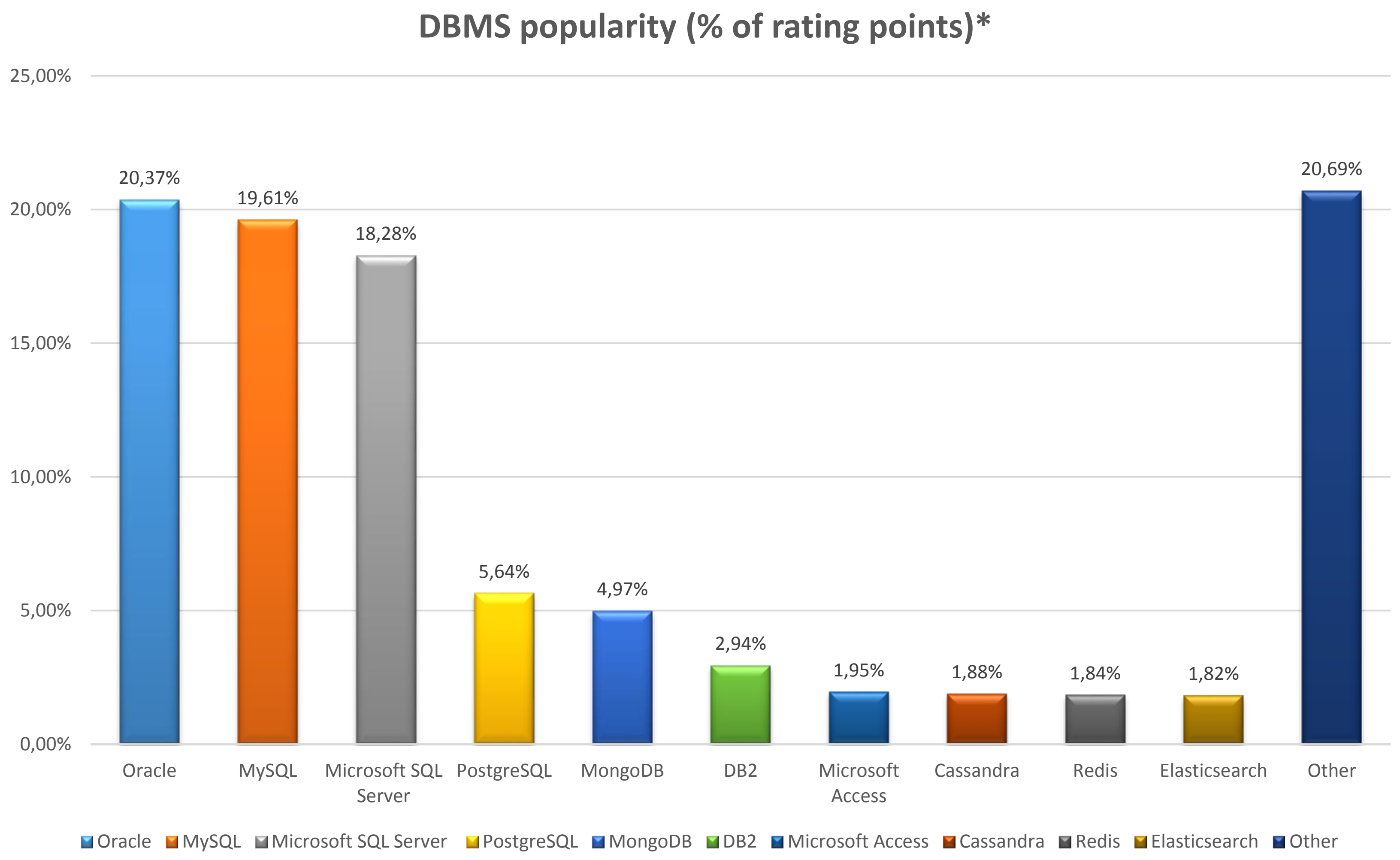
DBMS (Database Management Systems)

- **Functions of DBMS:**
  - Provides data independence
  - Concurrency control
  - Provides recovery services
  - Provides utility services
  - Provides a clear and logical view of the process that manipulates data

- **Advantages of DBMS:**
  - Segregation of application program
  - Minimal data duplicity
  - Easy retrieval of data
  - Reduced development time and maintenance needed

- **Examples:** MySql, Oracle,Microsoft SQL Server, Microsoft Access.

DBMS overview



*based on data from https://db-engines.com

Oracle

- Developer: Oracle
- Initial release 1980, current version 12
- Commercial license
- Implementation language C,C++
- Server operating system : Linux, OS X, Solaris, Windows
- SQL support
- Partitioning: tables can be distributed across several files
- Other features : Concurrency, Transaction concepts, Triggers
- version: 11*g* Express Edition (max db size=11GB,max RAM=1GB,single CPU)

MySql

- Relational DBMS
- Developer: Oracle
- Initial release 1995, current version 5.7.19, July 2017
- Open Source
- Implementation language C,C++
- Server operating system : Linux, Solaris, Windows, FreeBSD
- SQL support
- Partitioning: horizontal partitioning
- Other features : Concurrency, Transaction concepts, Triggers

SQL Server

- Relational DBMS
- Developer: Microsoft
- Initial release 1989, current version 2016
- Commercial license
- Implementation language C++
- Server operating system : Windows
- SQL support
- Partitioning: tables can be distributed across several files
- Other features : Concurrency, Transaction concepts, Triggers
- Freeware version: Express Edition (max db size=10GB,max RAM=1GB,single CPU)

PostgreSQL

- Relational DBMS
- Developer: PostgreSQL Global Development Group
- Initial release 1989, current version 9.6.5, August 2017
- Open Source
- Implementation language C
- Server operating system : Linux, Solaris, Windows
- SQL support
- Partitioning: no
- Other features : Concurrency, Transaction concepts, Triggers

MongoDB

- Relational DBMS
- Developer: MongoDB, Inc.
- initial release 2009, current version 3.4.9, September 2017
- Open Source
- implementation language C++
- Server operating system : Linux, Solaris, Windows
- SQL support
- Partitioning: Sharding
- Other features : Concurrency

SQL Introduction

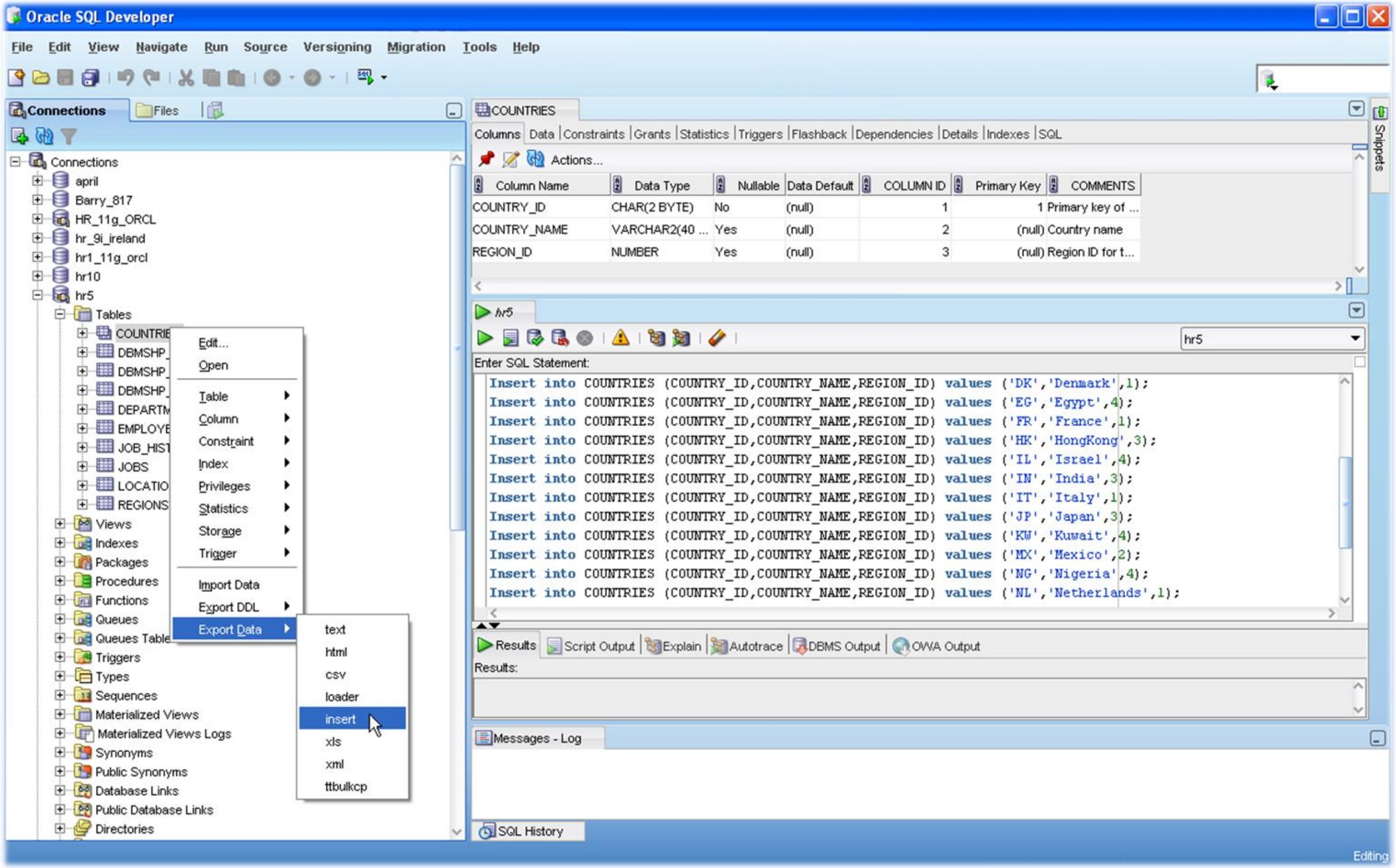
- **SQL -** Structure Query Language
- Language for generating, manipulating, and retrieving data from a relational database
- Developed at IBM by Donald D. Chamberlin and Raymond F. Boyce (early 70's)
- Initially known as Structured English Query Language (SEQUEL)
- In 1979 Oracle introduced the first commercially available implementation of SQL
- main features:
  - cross-platform (MySql, Oracle, MSQL,…)
  - easy to learn
  - allows the user to create, update, delete, and retrieve data from a database

Column Types

- When we design our database we need to decide what type of data will be stored in each column.
- Choosing appropriate data type is really important for optimized storage and speed of our database.
- Most data types are fixed and strict. Their narrow definition is what allows us to properly optimize storage and data manipulation.
- Additionally our columns can have following attributes:
  - Default value
  - Primary/Unique/Index/Fulltext Index
  - Auto Increment
  - And several other of lower importance.

DB AdminTool: Oracle SQL Developer (for Oracle)

- http://www.oracle.com/
- freeware



Column Name	Data Type	Nullable	Data Default	COLUMN ID	Primary Key	COMMENTS
COUNTRY_ID	CHAR(2 BYTE)	No	(null)	1	1	Primary key of ...
COUNTRY_NAME	VARCHAR2(40 ...	Yes	(null)	2	(null)	Country name
REGION_ID	NUMBER	Yes	(null)	3	(null)	Region ID for t...

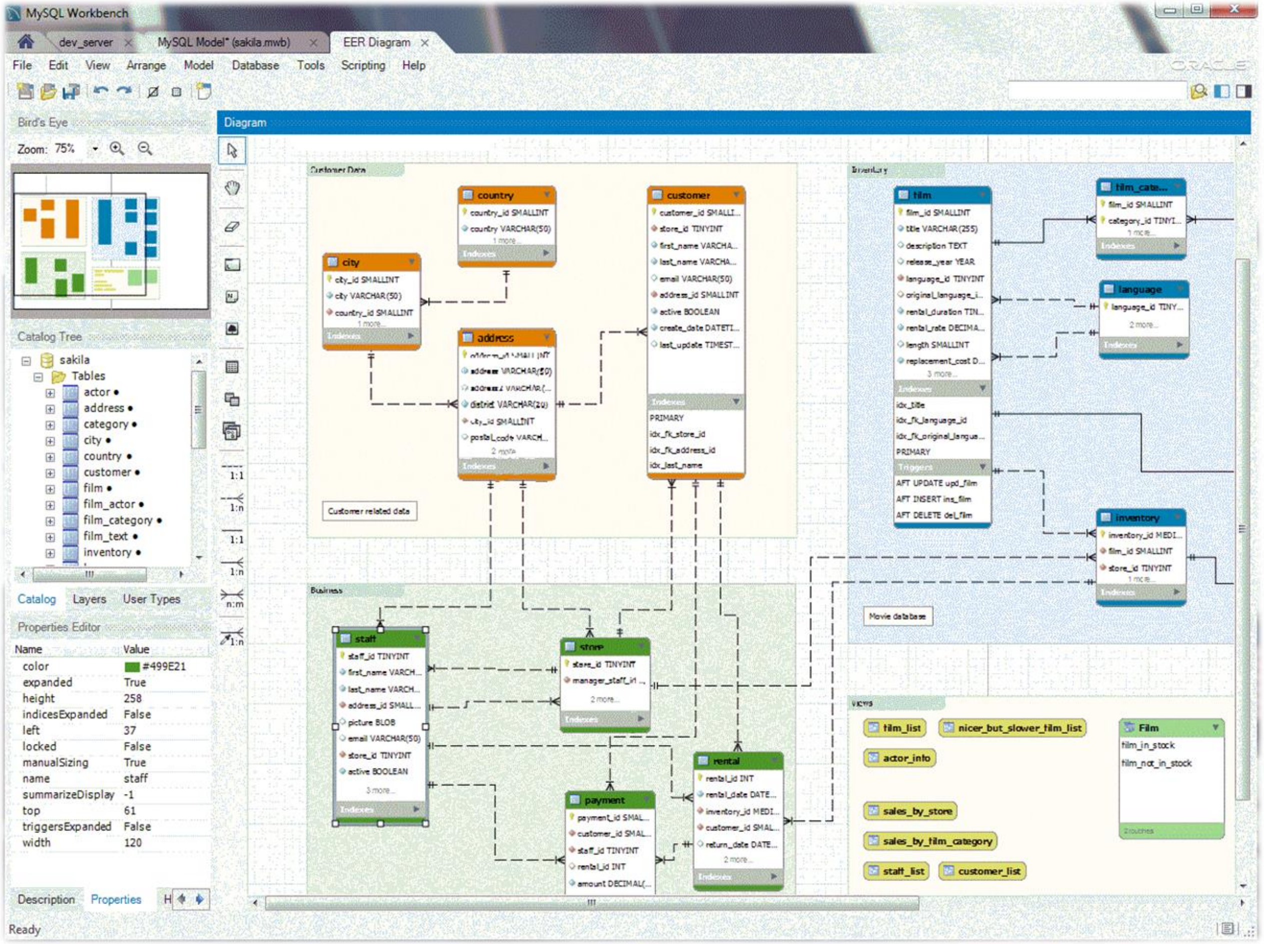
Enter SQL Statement:

```
Insert into COUNTRIES (COUNTRY_ID,COUNTRY_NAME,REGION_ID) values ('DK','Denmark',1);
Insert into COUNTRIES (COUNTRY_ID,COUNTRY_NAME,REGION_ID) values ('EG','Egypt',4);
Insert into COUNTRIES (COUNTRY_ID,COUNTRY_NAME,REGION_ID) values ('FR','France',1);
Insert into COUNTRIES (COUNTRY_ID,COUNTRY_NAME,REGION_ID) values ('HK','HongKong',3);
Insert into COUNTRIES (COUNTRY_ID,COUNTRY_NAME,REGION_ID) values ('IL','Israel',4);
Insert into COUNTRIES (COUNTRY_ID,COUNTRY_NAME,REGION_ID) values ('IN','India',3);
Insert into COUNTRIES (COUNTRY_ID,COUNTRY_NAME,REGION_ID) values ('IT','Italy',1);
Insert into COUNTRIES (COUNTRY_ID,COUNTRY_NAME,REGION_ID) values ('JP','Japan',3);
Insert into COUNTRIES (COUNTRY_ID,COUNTRY_NAME,REGION_ID) values ('KW','Kuwait',4);
Insert into COUNTRIES (COUNTRY_ID,COUNTRY_NAME,REGION_ID) values ('MX','Mexico',2);
Insert into COUNTRIES (COUNTRY_ID,COUNTRY_NAME,REGION_ID) values ('NG','Nigeria',4);
Insert into COUNTRIES (COUNTRY_ID,COUNTRY_NAME,REGION_ID) values ('NL','Netherlands',1);
```

Results:

DB AdminTool: MySQL Workbench (for MySQL)

- https://www.mysql.com/
- freeware



Name	Value
color	🟩 #499E21
expanded	True
height	258
indicesExpanded	False
left	37
locked	False
manualSizing	True
name	staff
summarizeDisplay	-1
top	61
triggersExpanded	False
width	120

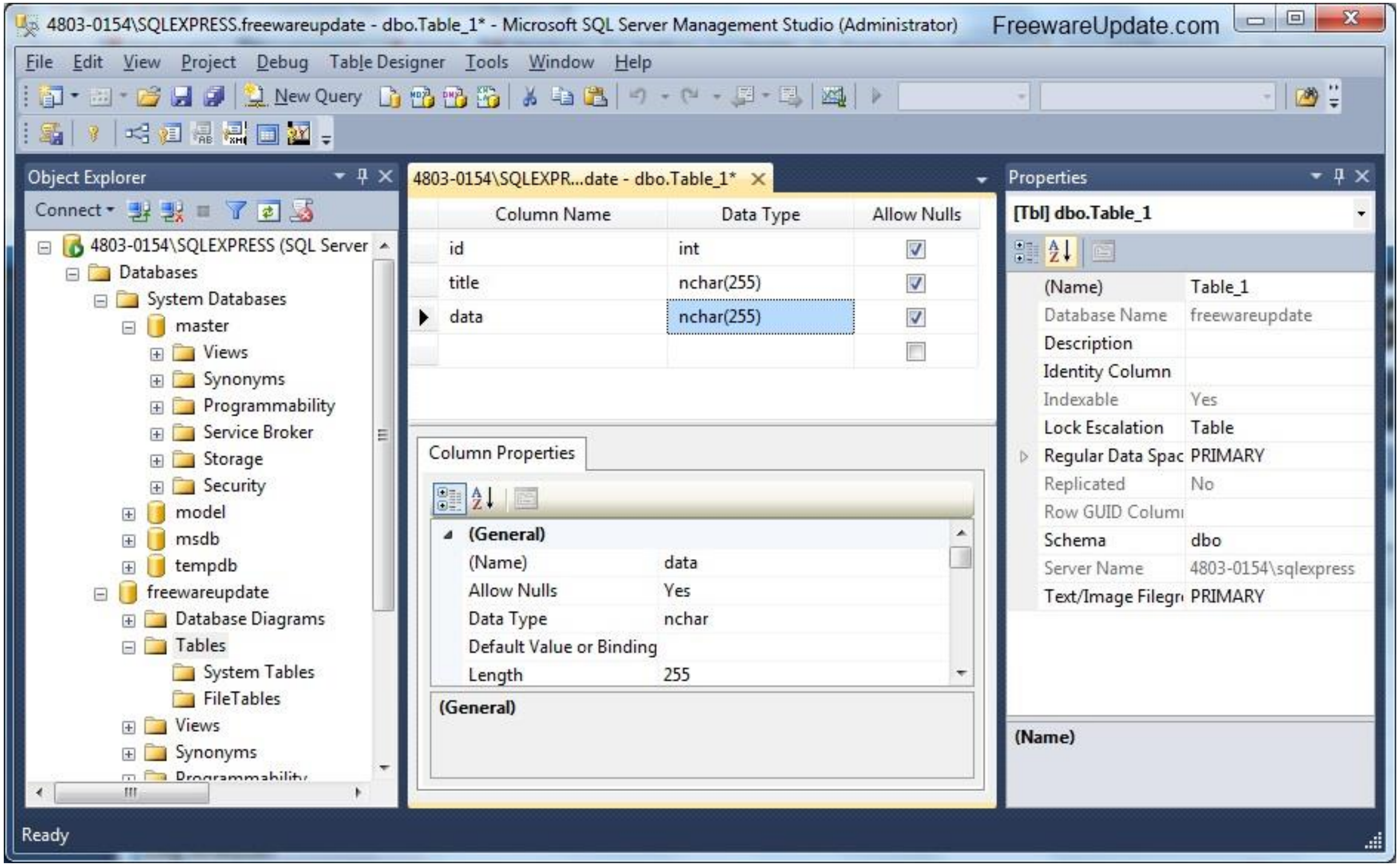
Extras:

-free MySQL host
https://www.db4free.net/signup.php

-free Workbench client
https://www.rollapp.com/app/mysqlworkbench

DB AdminTool: SQL Server Management Studio Express (for mSQL)

- https://www.microsoft.com
- freeware



Column Name	Data Type	Allow Nulls
id	int	☑
title	nchar(255)	☑
data	nchar(255)	☑
		□

Column Properties

(General)	
(Name)	data
Allow Nulls	Yes
Data Type	nchar
Default Value or Binding	
Length	255

(General)

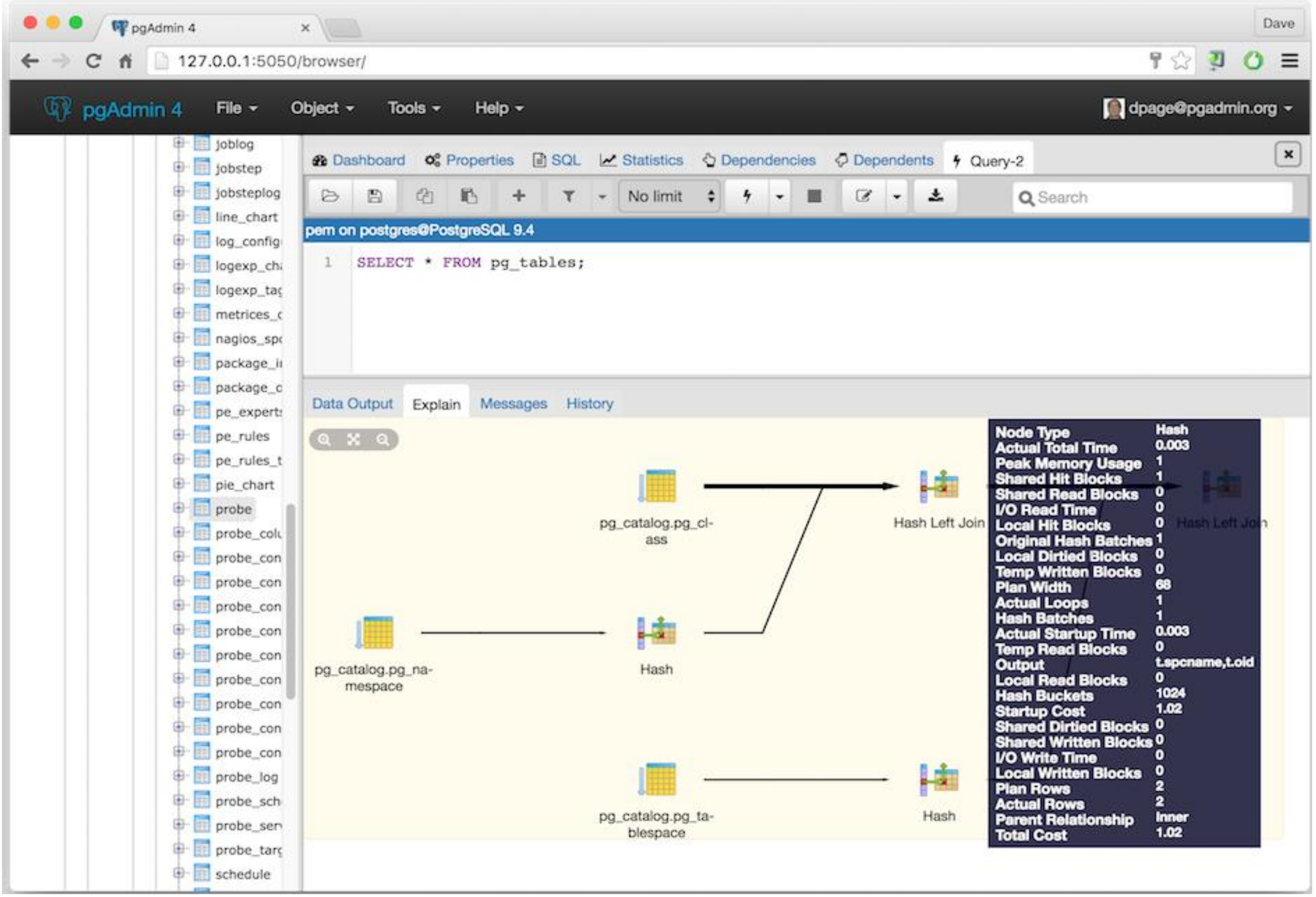
Properties: [Tbl] dbo.Table_1

(Name)	Table_1
Database Name	freewareupdate
Description	
Identity Column	
Indexable	Yes
Lock Escalation	Table
Regular Data Spac	PRIMARY
Replicated	No
Row GUID Colum	
Schema	dbo
Server Name	4803-0154\sqlexpress
Text/Image Filegr	PRIMARY

(Name)

DB AdminTool: pgAdmin (for PostgreSQL)

- https://www.postgresql.org
- freeware



pem on postgres@PostgreSQL 9.4

```
SELECT * FROM pg_tables;
```

Node Type	Hash
Actual Total Time	0.003
Peak Memory Usage	1
Shared Hit Blocks	1
Shared Read Blocks	0
I/O Read Time	0
Local Hit Blocks	0
Original Hash Batches	1
Local Dirtied Blocks	0
Temp Written Blocks	0
Plan Width	68
Actual Loops	1
Hash Batches	1
Actual Startup Time	0.003
Temp Read Blocks	0
Output	t.spcname,t.oid
Local Read Blocks	0
Hash Buckets	1024
Startup Cost	1.02
Shared Dirtied Blocks	0
Shared Written Blocks	0
I/O Write Time	0
Local Written Blocks	0
Plan Rows	2
Actual Rows	2
Parent Relationship	Inner
Total Cost	1.02

Column Types - part1 (MySql)

ColumnType	Description	Example of use
int	Stores integer (whole number) values in the range –2 147 483 648 to 2 147 483 647.	ex.: id int
decimal	Stores a fixed-point number.	ex.: price decimal(4,2) values from -99,99 to 99,99 4- total number of digits 2- number of digits after the comma
double	Stores floating-point numbers.	
float	Stores floating-point numbers.	
date	Stores and displays a date in the format YYYY-MM-DD for the range 1000-01-01 to 9999-12-31. Alternative formats: YYYY/MM/DD, YYYYMMDD	ex. : birth date
time	Stores a time in the format HHH:MM:SS for the range -838:59:59 to 838:59:59 Alternative formats: DD HH:MM:SS, HH:MM:SS, DD HH:MM, HH:MM,	

Column Types-part 2 (MySql)

ColumnType	Description	Example
Char	commonly used string type. Uses always as much space as declared. Char(10) means to allocate always 10 characters.	ex.: name char(10)
vachar	commonly used string type. Uses only as much space as you need, plus one byte to store the length of string.	ex.: name varchar(10)
text	used to store large string data objects. Text type stores a variable amount of data (such as a document or other text file) up to 65,535 bytes in length.	
blob	used for storing large data (binary format). Stores a variable amount of data (such as an image, video, or other nontext file) up to 65,535 bytes in length.	
enum	a list, or enumeration of string values.	ex.: fruit_name ENUM('Apple', 'Orange', 'Pear')

Column Types

- There are more data types. We encourage you to browse the full list at least once:
  - https://dev.mysql.com/doc/refman/5.7/en/data-types.html
  - https://www.w3schools.com/sql/sql_datatypes.asp
- Once again. Choosing appropriate data type is really important for optimized storage and speed of our database.

SQL Naming conventions

- Most important think with naming conventions is to be consistent across all tables as much as possible.
- There is no such think as „the best" naming convention
- Remember you database structure can outlast any application that uses it and many developers and end users.
- Names of tables should be short but meaningful and precise.
- In naming convention most important attributes are:
  - Plural vs singular
  - Capitalization
  - Spaces
  - Prefixes/suffixes
  - Primary key naming convention

SQL Naming conventions

- Plural vs Singular:
  - Some guidelines advocate plural names for tables and singular names for columns. Table Customers can have a column customerId
  - On the other hand using plurals for table names can be sometimes confusing (plural forms are not always straightforward)
  - Plural forms are more natural when read as natural language. We are taking a customer from table full of customers. However sometimes when we address a database table as an object a notation customer.customerId is also natural.

SQL Naming conventions

- There are many options in terms of capitalizations:
  - alllowrcase
  - ALLUPPERCASE
  - camelCase
  - PascalCase
- Additionally we need to decide if we want to use underscore for separation:
  - All_lower_case
  - ALL_UPPER_CALE
  - damel_Case
  - Pascel_Case

SQL Naming conventions

- Prefixes/suffixes and primary key naming conventions.
  - We can meet many accepted prefixes:
    - IX – index
    - PK – Primary Key
    - FK – Foreign Key
    - CK – Check Constraint
    - DF – Default
    - UQ – Unique
- Primary Keys are for us of special importance. Few of popular conventions are listed below:
  - id
  - <table_name>Id
  - Id<table_name>

SQL Naming conventions

- Remember there is no one best answer. You can read more here:
  - https://stackoverflow.com/questions/522356/what-sql-coding-standard-do-you-follow
  - http://www.vertabelo.com/blog/technical-articles/naming-conventions-in-database-modeling
  - http://www.vertabelo.com/blog/technical-articles/an-unemotional-logical-look-at-sql-server-naming-conventions
  - https://www.codeproject.com/Articles/1065295/SQL-Server-Table-and-Column-Naming-Conventions
  - http://leshazlewood.com/software-engineering/sql-style-guide/

Student’s Task (Database design)

- **basing on data below and using normalization rules, please design database at https://api.genmymodel.com/**

Name	Surname	DateOfBirth	HomeAddress	Employer	WorkedYears
Tom	Gold	1980-08-01	Silver Street 13, flat 12,Small City, 02-344	MircoTech	5
Tom	Gold	1980-08-01	Silver Street 13, flat 12,Small City, 02-344	BigTech	10
Ann	Smart	1960-06-07	Cambridge Street 11,Green City,04-233	Zara	3
Ann	Smart	1960-06-07	Cambridge Street 11,Green City,04-233	Nike	5
Alex	Brown	1964-06-07	Oxford Street 11,London City,00-231	EY	10

References

1. **Williams H., S. Tahaghoghi (2009). Learning MySQL., O'Reilly Media**
2. **Churcher C. (2007), Beginning Database Design.,Apress**