# Databases in Biomedical Sciences

Kei Cheung, Ph.D.

Professor

Department of Emergency Medicine

Yale Center for Medical Informatics

(2/23/2022)

# The 4th paradigm: data-intensive scientific discovery

- It expands the vision of Jim Gray (Mr. Database)
- His vision of a Personal Memex as well as a World Memex
  - Memex (originally coined by Vannevar Bush in 1945) is a device in which an individual stores all his books, records, and communications

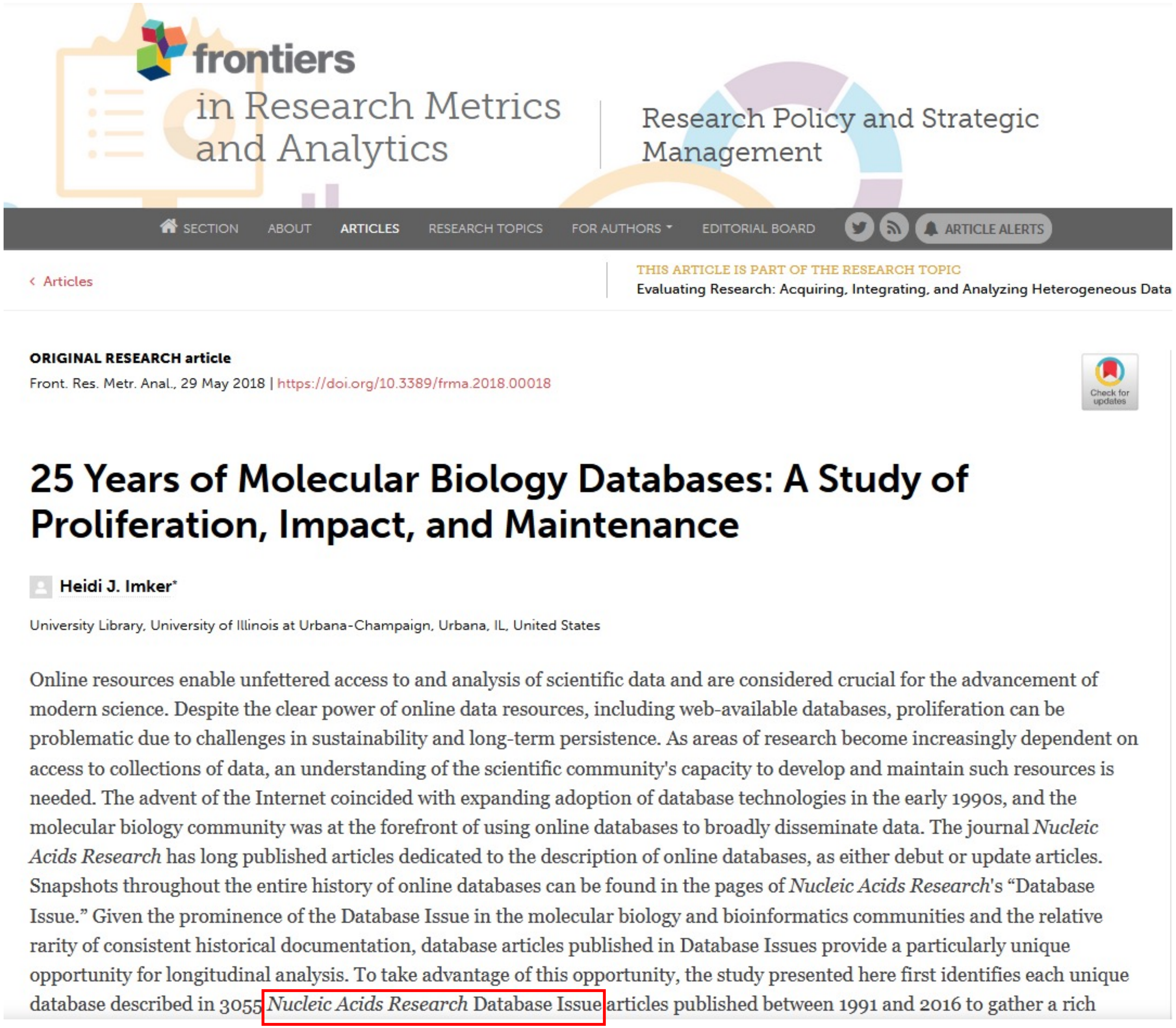
**Received the Turing Award in 1998**
**Disappeared at sea in 2007**

# Healthcare and life sciences data sources

Drug Research
Social Media
Patient Records
Gene Sequencing
Test Results
Claims
Home Monitoring
Mobile Apps

4Vs:

- Volume – high-throughput technologies
- Variety – diverse data types, different formats, structured vs. unstructured data
- Velocity – data streaming
- Veracity – trust worthiness of data



THIS ARTICLE IS PART OF THE RESEARCH TOPIC
Evaluating Research: Acquiring, Integrating, and Analyzing Heterogeneous Data

ORIGINAL RESEARCH article

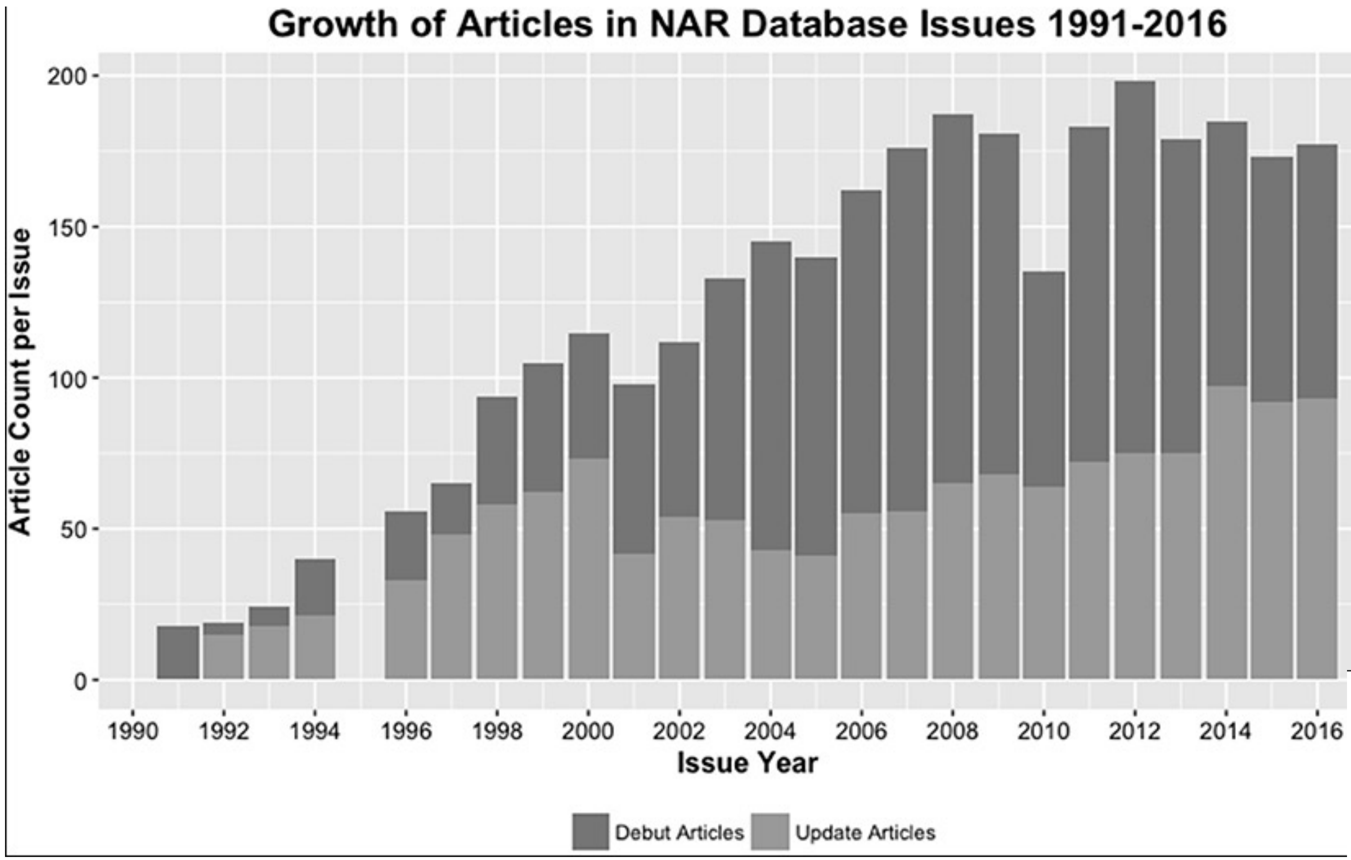
Front. Res. Metr. Anal., 29 May 2018 | https://doi.org/10.3389/frma.2018.00018

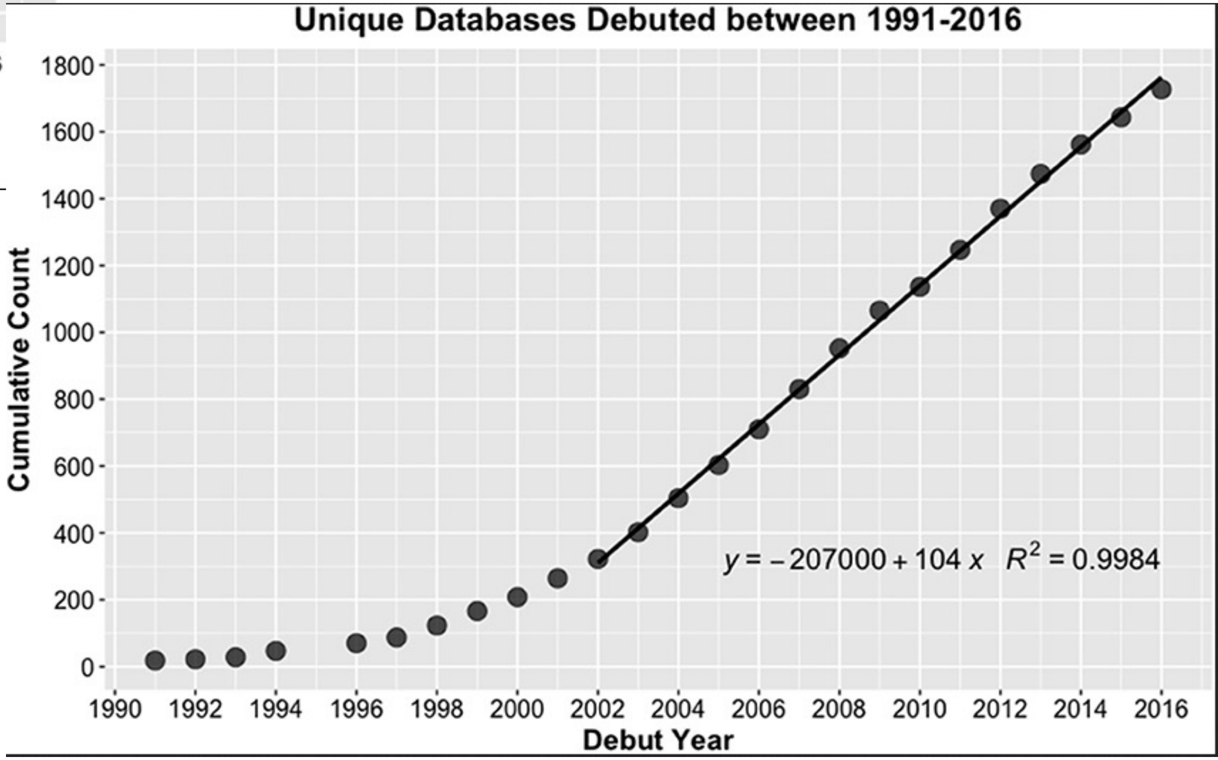
# 25 Years of Molecular Biology Databases: A Study of Proliferation, Impact, and Maintenance

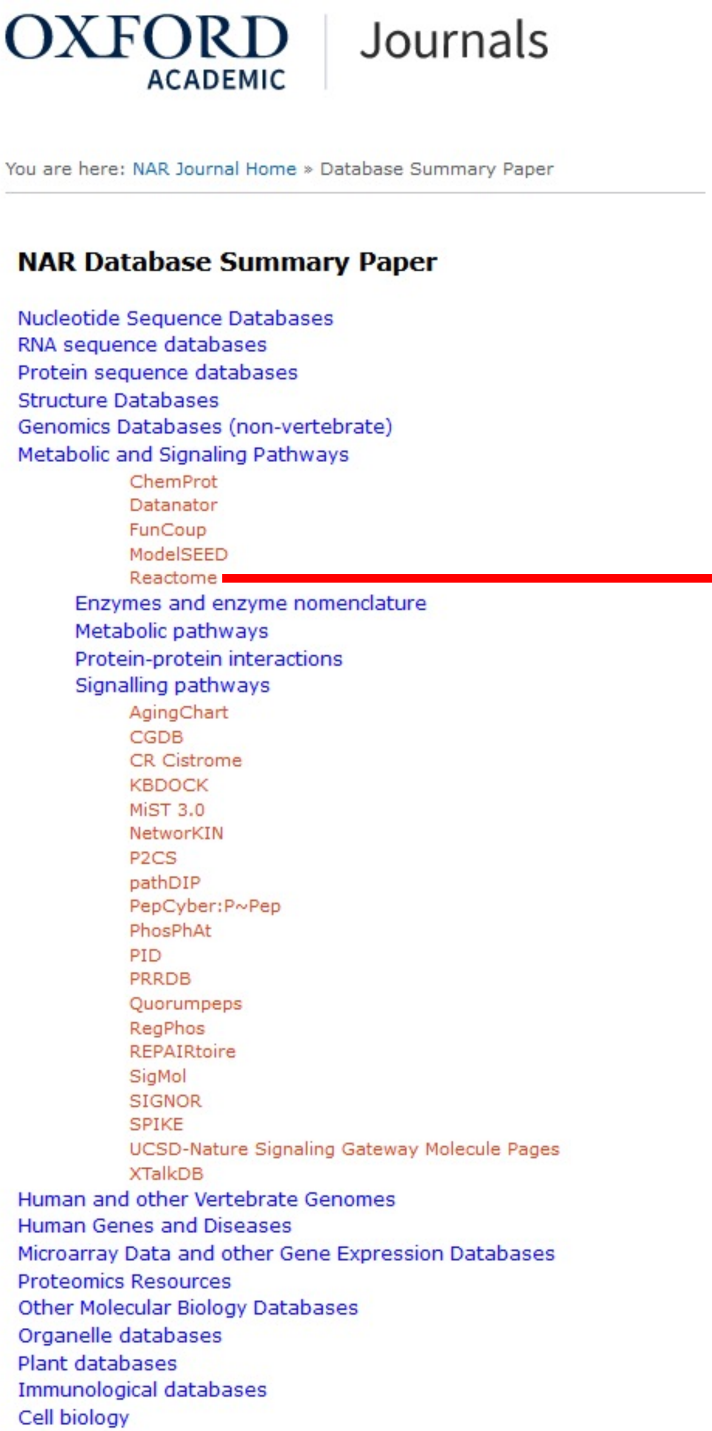
Heidi J. Imker*

University Library, University of Illinois at Urbana-Champaign, Urbana, IL, United States

Online resources enable unfettered access to and analysis of scientific data and are considered crucial for the advancement of modern science. Despite the clear power of online data resources, including web-available databases, proliferation can be problematic due to challenges in sustainability and long-term persistence. As areas of research become increasingly dependent on access to collections of data, an understanding of the scientific community's capacity to develop and maintain such resources is needed. The advent of the Internet coincided with expanding adoption of database technologies in the early 1990s, and the molecular biology community was at the forefront of using online databases to broadly disseminate data. The journal *Nucleic Acids Research* has long published articles dedicated to the description of online databases, as either debut or update articles. Snapshots throughout the entire history of online databases can be found in the pages of *Nucleic Acids Research*'s "Database Issue." Given the prominence of the Database Issue in the molecular biology and bioinformatics communities and the relative rarity of consistent historical documentation, database articles published in Database Issues provide a particularly unique opportunity for longitudinal analysis. To take advantage of this opportunity, the study presented here first identifies each unique database described in 3055 *Nucleic Acids Research* Database Issue articles published between 1991 and 2016 to gather a rich


Growth of Articles in NAR Database Issues 1991-2016
Article Count per Issue
0
50
100
150
200
1990
1992
1994
1996
1998
2000
2002
2004
2006
2008
2010
2012
2014
2016
Issue Year
Debut Articles
Update Articles

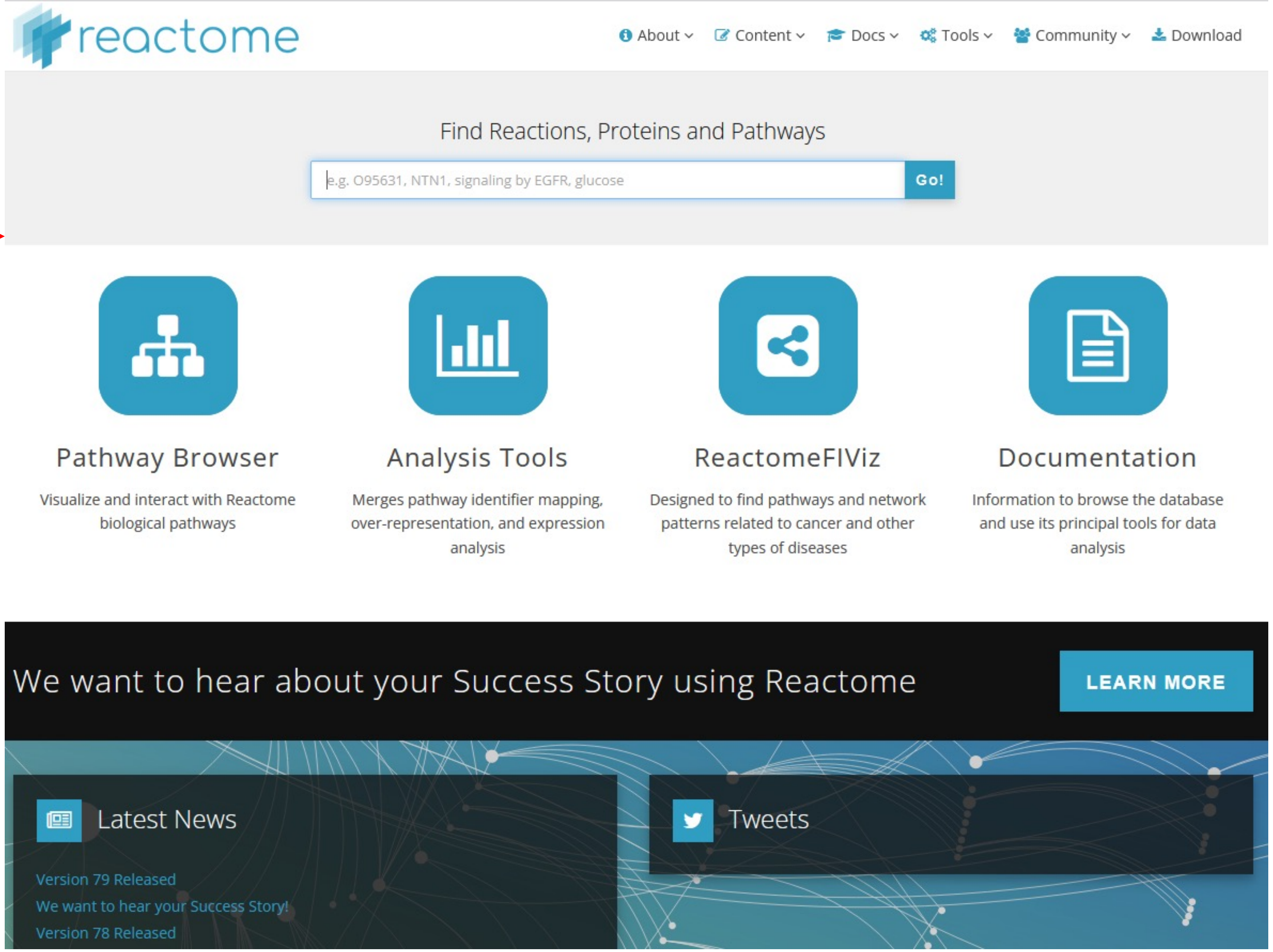

Unique Databases Debuted between 1991-2016
Cumulative Count
0
200
400
600
800
1000
1200
1400
1600
1800
$y = -207000 + 104\,x \quad R^2 = 0.9984$
1990
1992
1994
1996
1998
2000
2002
2004
2006
2008
2010
2012
2014
2016
Debut Year



You are here: NAR Journal Home » Database Summary Paper

## NAR Database Summary Paper

- Nucleotide Sequence Databases
- RNA sequence databases
- Protein sequence databases
- Structure Databases
- Genomics Databases (non-vertebrate)
- Metabolic and Signaling Pathways
    - ChemProt
    - Datanator
    - FunCoup
    - ModelSEED
    - Reactome
  - Enzymes and enzyme nomenclature
  - Metabolic pathways
  - Protein-protein interactions
  - Signalling pathways
    - AgingChart
    - CGDB
    - CR Cistrome
    - KBDOCK
    - MiST 3.0
    - NetworKIN
    - P2CS
    - pathDIP
    - PepCyber:P~Pep
    - PhosPhAt
    - PID
    - PRRDB
    - Quorumpeps
    - RegPhos
    - REPAIRtoire
    - SigMol
    - SIGNOR
    - SPIKE
    - UCSD-Nature Signaling Gateway Molecule Pages
    - XTalkDB
- Human and other Vertebrate Genomes
- Human Genes and Diseases
- Microarray Data and other Gene Expression Databases
- Proteomics Resources
- Other Molecular Biology Databases
- Organelle databases
- Plant databases
- Immunological databases
- Cell biology



Find Reactions, Proteins and Pathways

**Pathway Browser**
Visualize and interact with Reactome biological pathways

**Analysis Tools**
Merges pathway identifier mapping, over-representation, and expression analysis

**ReactomeFIViz**
Designed to find pathways and network patterns related to cancer and other types of diseases

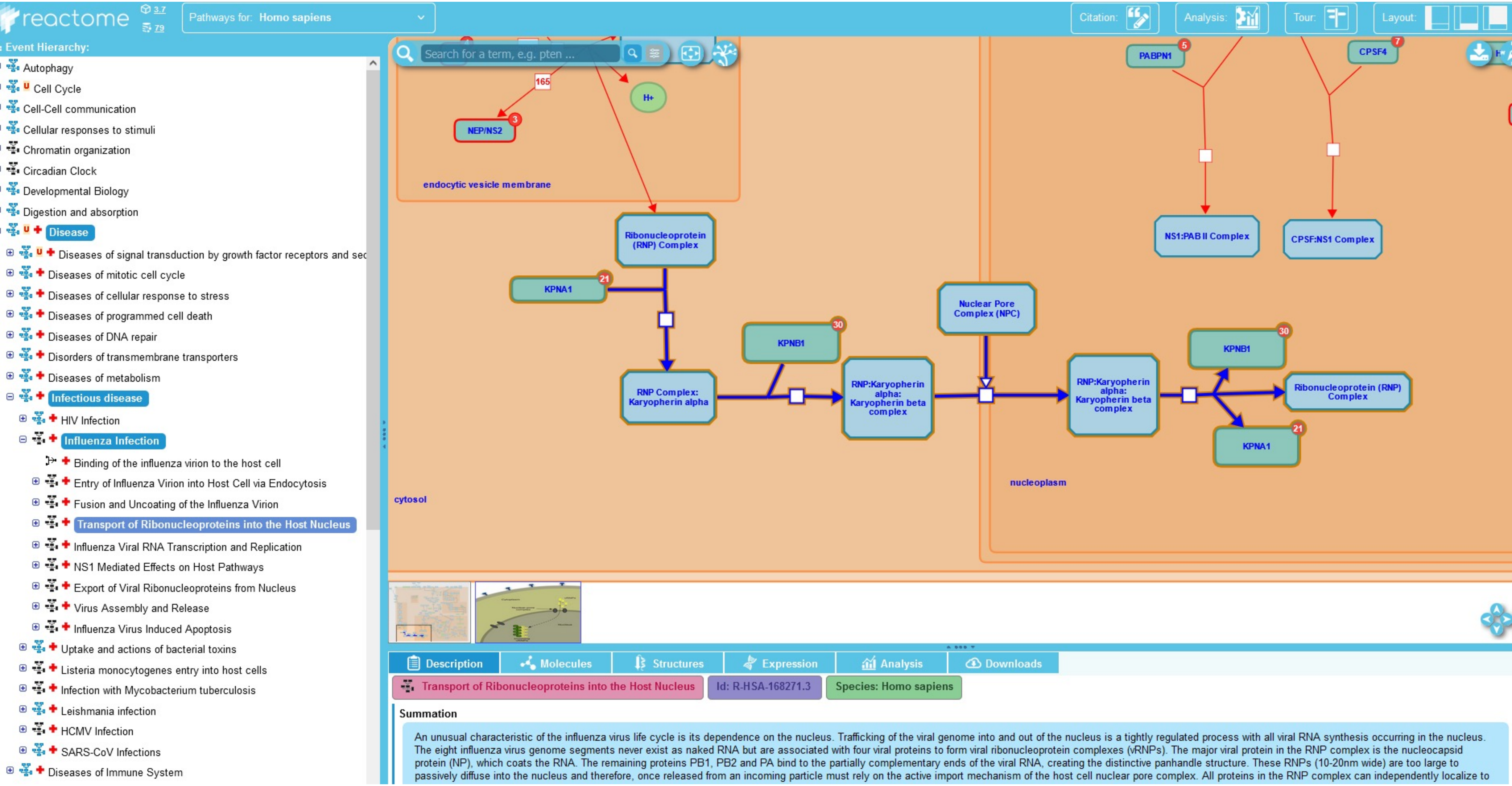
**Documentation**
Information to browse the database and use its principal tools for data analysis

We want to hear about your Success Story using Reactome

Latest News

- Version 79 Released
- We want to hear your Success Story!
- Version 78 Released

Tweets


reactome
3.7
79
Pathways for: Homo sapiens
Citation:
Analysis:
Tour:
Layout:
Event Hierarchy:
Autophagy
Cell Cycle
Cell-Cell communication
Cellular responses to stimuli
Chromatin organization
Circadian Clock
Developmental Biology
Digestion and absorption
Disease
Diseases of signal transduction by growth factor receptors and sec
Diseases of mitotic cell cycle
Diseases of cellular response to stress
Diseases of programmed cell death
Diseases of DNA repair
Disorders of transmembrane transporters
Diseases of metabolism
Infectious disease
HIV Infection
Influenza Infection
Binding of the influenza virion to the host cell
Entry of Influenza Virion into Host Cell via Endocytosis
Fusion and Uncoating of the Influenza Virion
Transport of Ribonucleoproteins into the Host Nucleus
Influenza Viral RNA Transcription and Replication
NS1 Mediated Effects on Host Pathways
Export of Viral Ribonucleoproteins from Nucleus
Virus Assembly and Release
Influenza Virus Induced Apoptosis
Uptake and actions of bacterial toxins
Listeria monocytogenes entry into host cells
Infection with Mycobacterium tuberculosis
Leishmania infection
HCMV Infection
SARS-CoV Infections
Diseases of Immune System
Search for a term, e.g. pten ...
165
NEP/NS2
3
H+
endocytic vesicle membrane
PABPN1
5
CPSF4
7
NS1:PAB II Complex
CPSF:NS1 Complex
Ribonucleoprotein (RNP) Complex
KPNA1
21
RNP Complex: Karyopherin alpha
KPNB1
30
RNP:Karyopherin alpha: Karyopherin beta complex
Nuclear Pore Complex (NPC)
RNP:Karyopherin alpha: Karyopherin beta complex
KPNB1
30
Ribonucleoprotein (RNP) Complex
KPNA1
21
cytosol
nucleoplasm
Description
Molecules
Structures
Expression
Analysis
Downloads

Transport of Ribonucleoproteins into the Host Nucleus

Id: R-HSA-168271.3

Species: Homo sapiens

Summation

An unusual characteristic of the influenza virus life cycle is its dependence on the nucleus. Trafficking of the viral genome into and out of the nucleus is a tightly regulated process with all viral RNA synthesis occurring in the nucleus. The eight influenza virus genome segments never exist as naked RNA but are associated with four viral proteins to form viral ribonucleoprotein complexes (vRNPs). The major viral protein in the RNP complex is the nucleocapsid protein (NP), which coats the RNA. The remaining proteins PB1, PB2 and PA bind to the partially complementary ends of the viral RNA, creating the distinctive panhandle structure. These RNPs (10-20nm wide) are too large to passively diffuse into the nucleus and therefore, once released from an incoming particle must rely on the active import mechanism of the host cell nuclear pore complex. All proteins in the RNP complex can independently localize to

# What is (not) a database?

- It’s not just a file
- It’s not just an Excel spreadsheet
- It’s an organized collection of related information that can easily be accessed, managed, and updated

# Difference between Spreadsheet and Database

| **Spreadsheet** | **Database** |
|---|---|
| Data analysis | Data management |
| Mathematical calculation | Structuring data and querying data to create subsets |
| Typically single user | Database management with multiple users |
| Formatting and chart display | Reports for data summarization |
| Limited in scale | Scalable |

↓

| | |
|---|---|
| Worksheet size: | 1,048,576 rows by 16,384 columns |
| Column width: | 255 characters |
| Total no. of characters that a cell can have: | 32,767 characters |

# Some key database concepts

- **Data integrity** is the assurance that data are correct and consistent (data correctly reflects the real world)
- **Data redundancy** occurs if data are duplicated between files
- **Data dependency** defines linkage between data files and their order of entry
- **Data security** refers to data being protected so that only authorized personnel can access them

# Relational database (SQL database)

- The relational model was introduced by E.F. Codd in 1970, which is based on the mathematical set theory
- A relational database management system (RDBMS) is a computer application (software) of the relational data model (e.g., MS SQLServer, MySQL, Oracle, …)
- It has become an industry standard with a standard query language (SQL)
- Relational databases have widely been used to manage data in different domains

# Components of Relational Database

- A table (relation) represents some class of objects (e.g., patients, doctors, drugs, hospitals)
- Each table consists of columns (attributes) and rows (tuples).
  - Each column represents some attribute of the object represented by the table (e.g., patient id, patient name)
  - Each row corresponds to an instance of the object represented by the table (e.g., each row in the Patient table represents a patient who has a specific patient id and name.)

# How to organize data into tables

# Keys

- Primary key: Every table should have a primary key comprising a single or multiple columns that contain unique values. A primary key is the unique identifier of a table row (e.g., “sample id” is the primary key for the **Sample** table)
- Foreign key: it is a key taken from a different table. For example, in the **Experiment** table, the “sample id” is the foreign key to the **Sample** table.

# Addition, Deletion and Modification Anomalies

| Student ID | Name | Address | Subject |
|---|---|---|---|
| 401 | Adam | Noida | Biology |
| 402 | Alex | Panipat | Math |
| 403 | Stuart | Jammu | Math |
| 404 | Adam | Noida | Physics |

# Normalization

- Normalization is a *process* in which we systematically organize columns and tables to eliminate anomalies due to data redundancy
- It involves decomposing a (de-normalized) table into less redundant (smaller) tables without losing information
- The objective is to isolate data so that additions, deletions, modifications of data can be made in just one table and then propagated to other tables using foreign keys.
- Normalization is a trade-off between data redundancy and performance.
  - Normalizing a table reduces data redundancy but introduces the need for joins when all of the data is required for a report query.
- **Normal Form**: A set of tables free from a certain set of addition, deletion and modification anomalies.

# Different Normal Forms

- **First normal form (1NF)**
- **Second normal form (2NF)**
- **Third normal form (3NF)**
- Boyce-Codd normal form (BCNF)
- Fourth normal form (4NF)
- Fifth normal form (5NF)
- Domain-Key normal form (DK/NF)
- …

# First Normal Form

- Each column value must be a single value only.
- All values for a given column must be of the same data type.
- Each column name must be unique.
- The order of columns is insignificant
- The order of the rows is insignificant
- No two rows in a table can be identical.

# First Normal Form Example

| ID | Student | Age | Subject |
|---|---|---|---|
| 401 | Adam | 15 | Biology |
| 404 | Adam | 15 | Physics |
| 402 | Alex | 14 | Math |
| 403 | Stuart | 17 | Math |

# Second Normal Form

- A table is in second normal form (2NF) if it is in 1NF and if all of its non-key columns are dependent on all of the *key*.
  - A table is in second normal form if it is free from partial-key dependencies
- Tables that have a single column for a key are automatically in 2NF.
  - This is one reason why we often use artificial identifiers (non-composite keys) as keys.
- To achieve second normal form, we may need to split a table into multiple tables and match rows between tables using primary and foreign keys

# Second Normal Form Example

| Student | Age |
|---|---|
| Adam | 15 |
| Alex | 14 |
| Stuart | 17 |

| Enroll_id | Student | Subject |
|---|---|---|
| 1 | Adam | Biology |
| 2 | Adam | Physics |
| 3 | Alex | Math |
| 4 | Stuart | Math |

# Third Normal Form

- Every non-primary key column must be dependent on primary key
- There should not be the case that a non-primary key column is determined by another non-primary key (*transitive dependency)*
  - Student (ID, Name, DOB, City, State, Zip)
- *A table is in 3NF if the following are true:*
  - *it is in 2NF*
  - *All transitive dependencies* are removed

Student (ID, Name, DOB, Zip)

Address (Zip, City, State)

# Entity Relationship Diagram (ERD)

# What is ERD

- It is a data model associated with a diagrammatic method (P. Chen 1976) used to conduct/view data modeling
- It describes the attributes of and the relationship between entities (data objects)
- DBA uses ERD to perform data modeling and explain the diagram to stakeholders

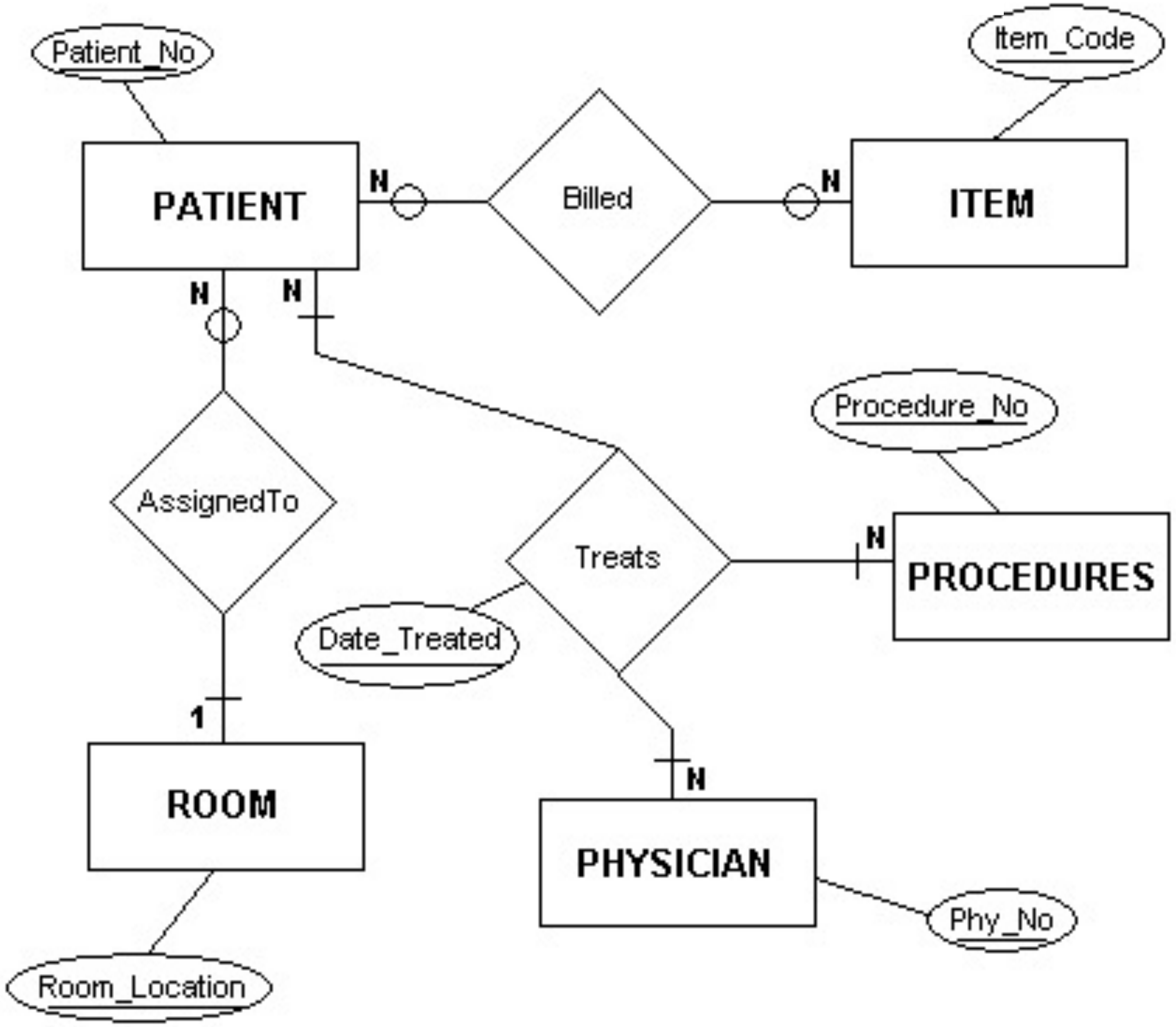
# Primary Components of ERD

- **Entity** represents a collection of objects in the real world (e.g., person, place, event)
- **Attribute** is a named property or characteristic of an entity
- **Relationship** is an association between the instances of one or more entities

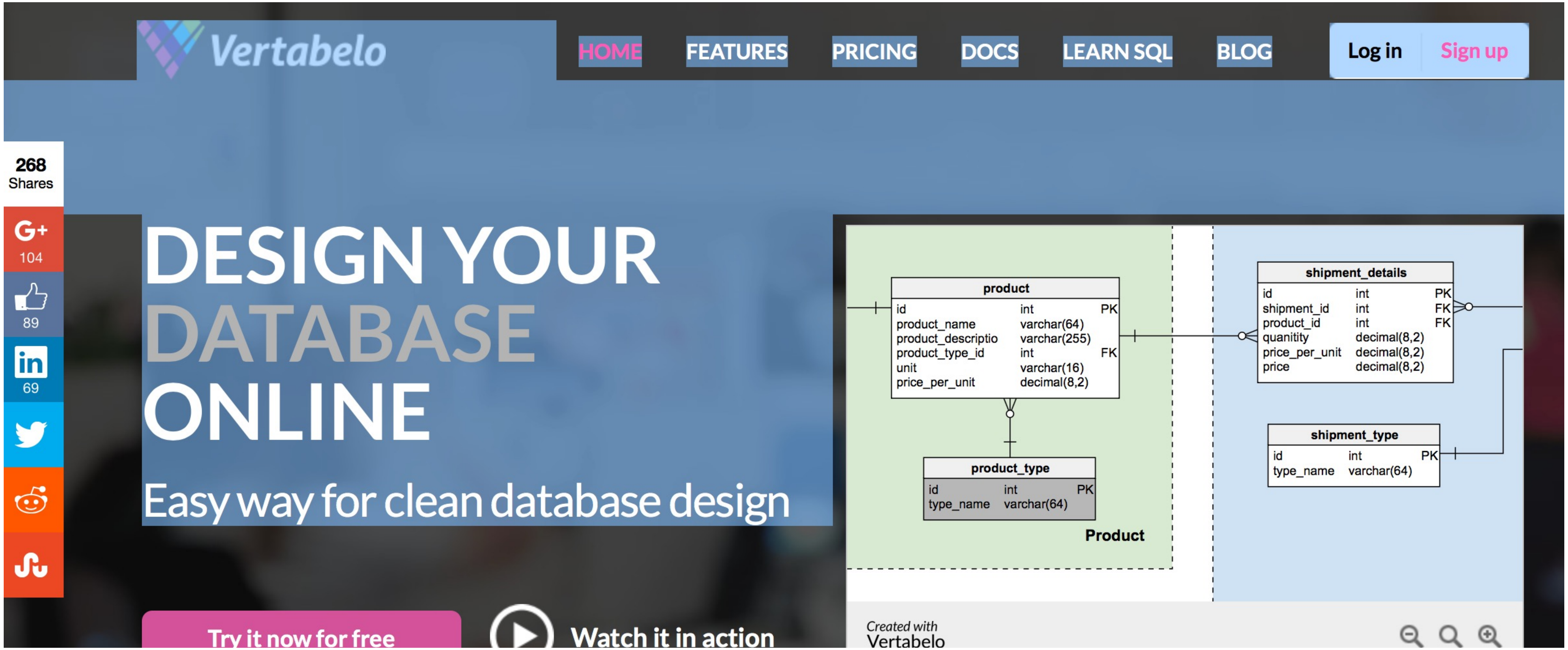
# Relationship Cardinality

- It expresses the minimum and maximum number of occurrences of one entity for a single occurrence of the other
  - One-to-One (1:1)
  - One-to-Many (1:N)
  - Many-to-Many (M:N)

# Example ERD (Hospital Database)



# Vertabelo (https://www.vertabelo.com/)



DESIGN YOUR DATABASE ONLINE

Easy way for clean database design

**product**

| | | |
|---|---|---|
| id | int | PK |
| product_name | varchar(64) | |
| product_descriptio | varchar(255) | |
| product_type_id | int | FK |
| unit | varchar(16) | |
| price_per_unit | decimal(8,2) | |

**product_type**

| | | |
|---|---|---|
| id | int | PK |
| type_name | varchar(64) | |

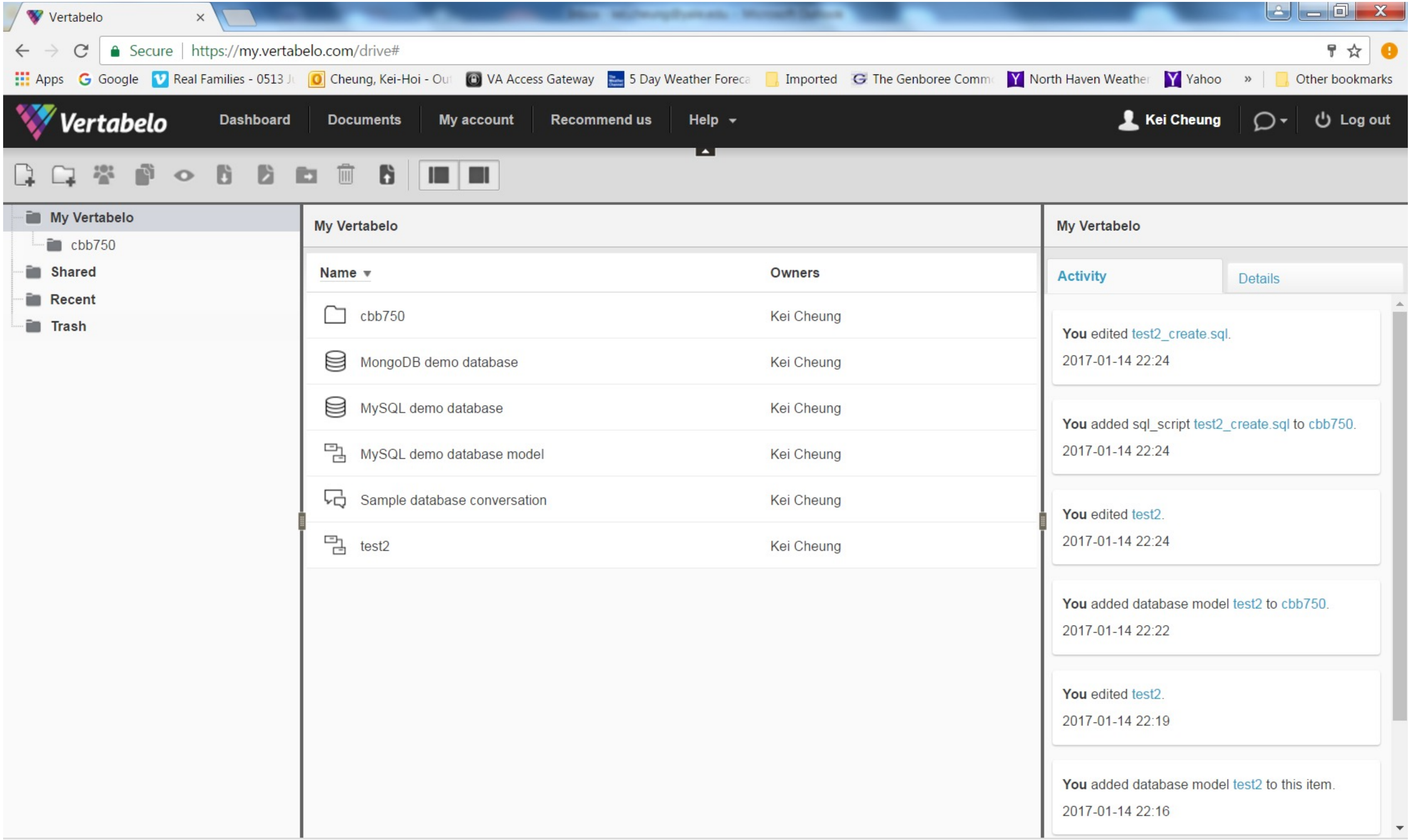
**Product**

**shipment_details**

| | | |
|---|---|---|
| id | int | PK |
| shipment_id | int | FK |
| product_id | int | FK |
| quanitity | decimal(8,2) | |
| price_per_unit | decimal(8,2) | |
| price | decimal(8,2) | |

**shipment_type**

| | | |
|---|---|---|
| id | int | PK |
| type_name | varchar(64) | |



My Vertabelo

| Name ▾ | Owners |
|---|---|
| cbb750 | Kei Cheung |
| MongoDB demo database | Kei Cheung |
| MySQL demo database | Kei Cheung |
| MySQL demo database model | Kei Cheung |
| Sample database conversation | Kei Cheung |
| test2 | Kei Cheung |

My Vertabelo

**You** edited test2_create.sql.
2017-01-14 22:24

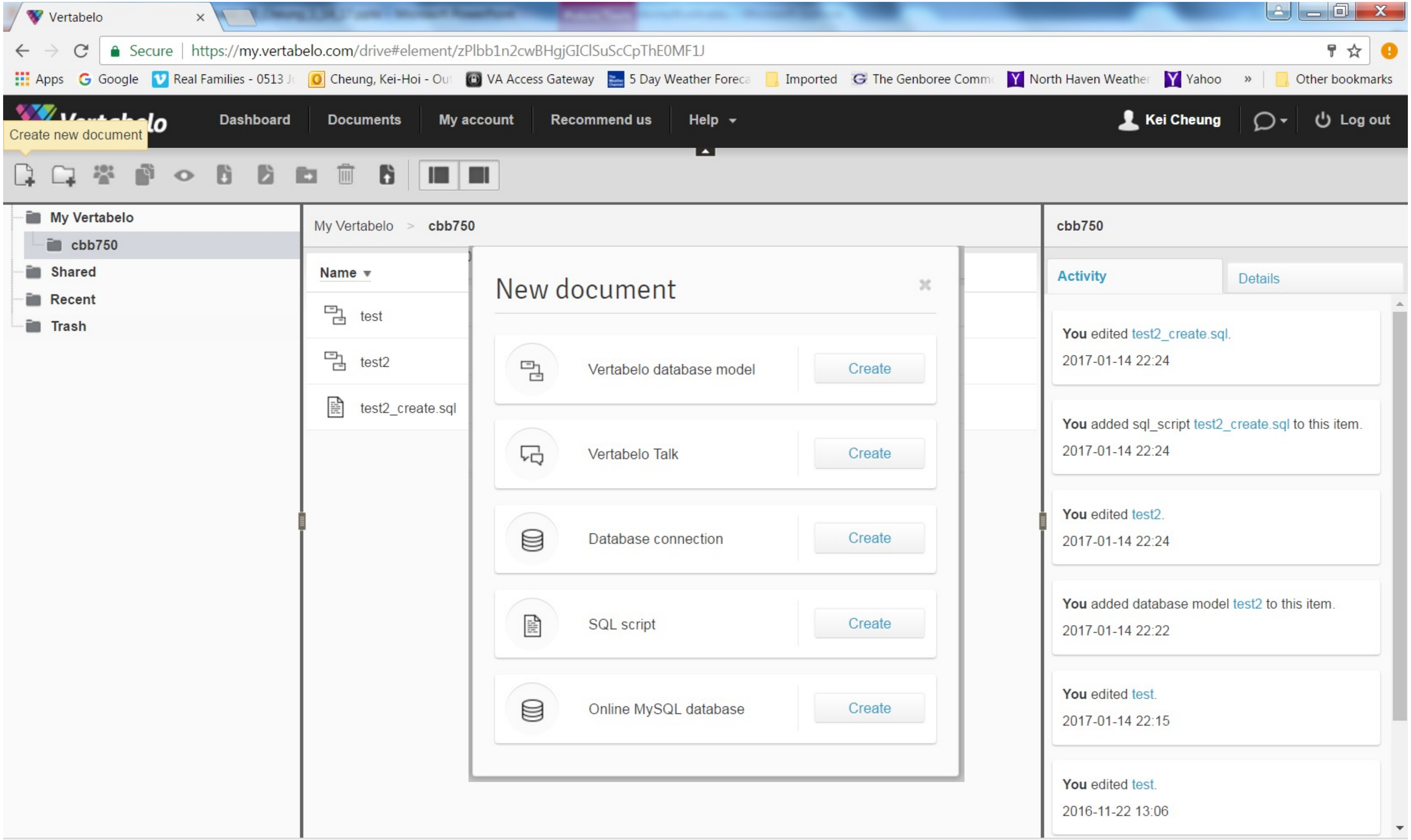
**You** added sql_script test2_create.sql to cbb750.
2017-01-14 22:24

**You** edited test2.
2017-01-14 22:24

**You** added database model test2 to cbb750.
2017-01-14 22:22

**You** edited test2.
2017-01-14 22:19

**You** added database model test2 to this item.
2017-01-14 22:16



New document

| | |
|---|---|
| Vertabelo database model | Create |
| Vertabelo Talk | Create |
| Database connection | Create |
| SQL script | Create |
| Online MySQL database | Create |

**You** edited test2_create.sql.
2017-01-14 22:24

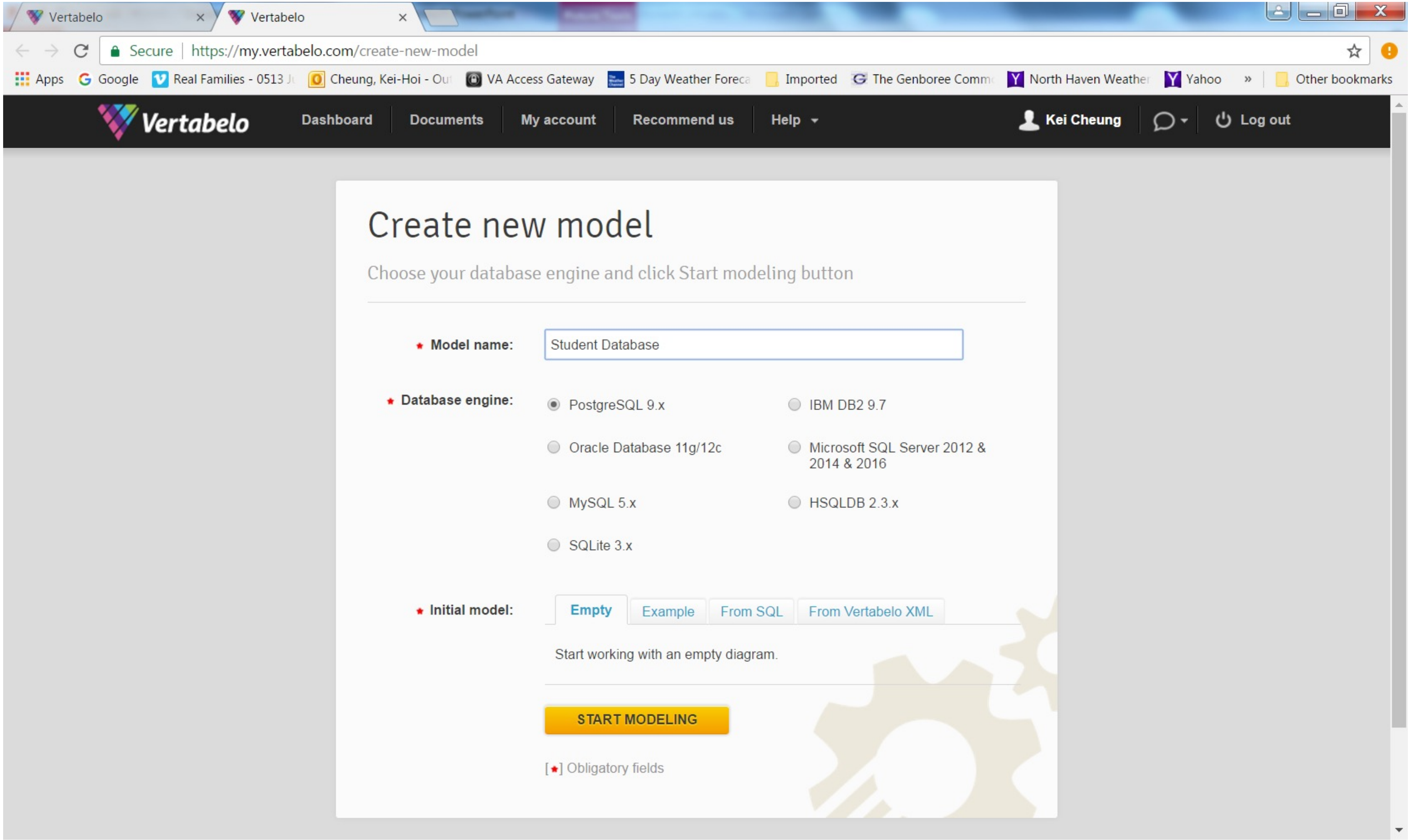
**You** added sql_script test2_create.sql to this item.
2017-01-14 22:24

**You** edited test2.
2017-01-14 22:24

**You** added database model test2 to this item.
2017-01-14 22:22

**You** edited test.
2017-01-14 22:15

**You** edited test.
2016-11-22 13:06



# Create new model

Choose your database engine and click Start modeling button

* Model name: Student Database

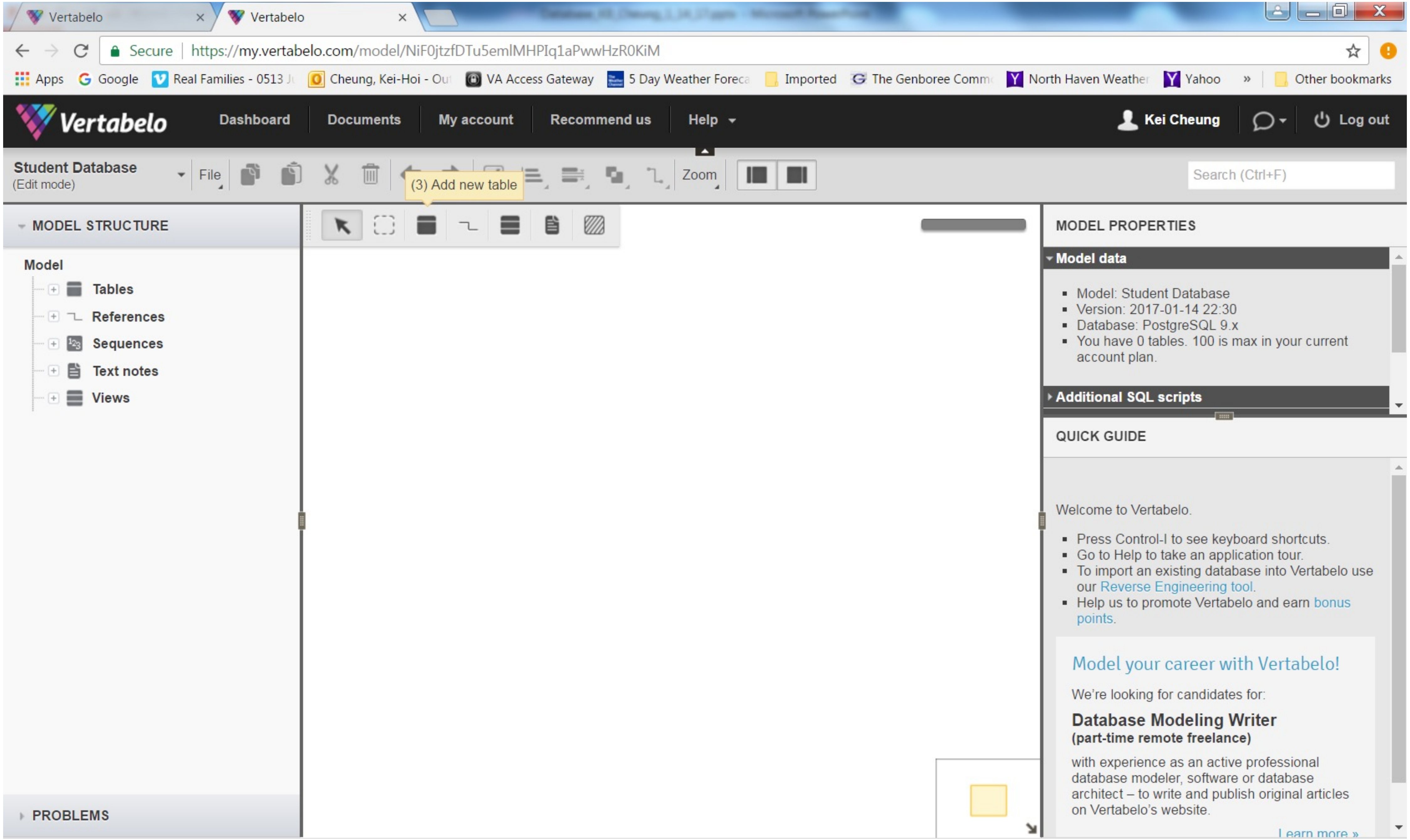
* Database engine:

● PostgreSQL 9.x
○ IBM DB2 9.7
○ Oracle Database 11g/12c
○ Microsoft SQL Server 2012 & 2014 & 2016
○ MySQL 5.x
○ HSQLDB 2.3.x
○ SQLite 3.x

* Initial model: Empty

Start working with an empty diagram.

[*] Obligatory fields



- Model: Student Database
- Version: 2017-01-14 22:30
- Database: PostgreSQL 9.x
- You have 0 tables. 100 is max in your current account plan.

Welcome to Vertabelo.

- Press Control-I to see keyboard shortcuts.
- Go to Help to take an application tour.
- To import an existing database into Vertabelo use our Reverse Engineering tool.
- Help us to promote Vertabelo and earn bonus points.

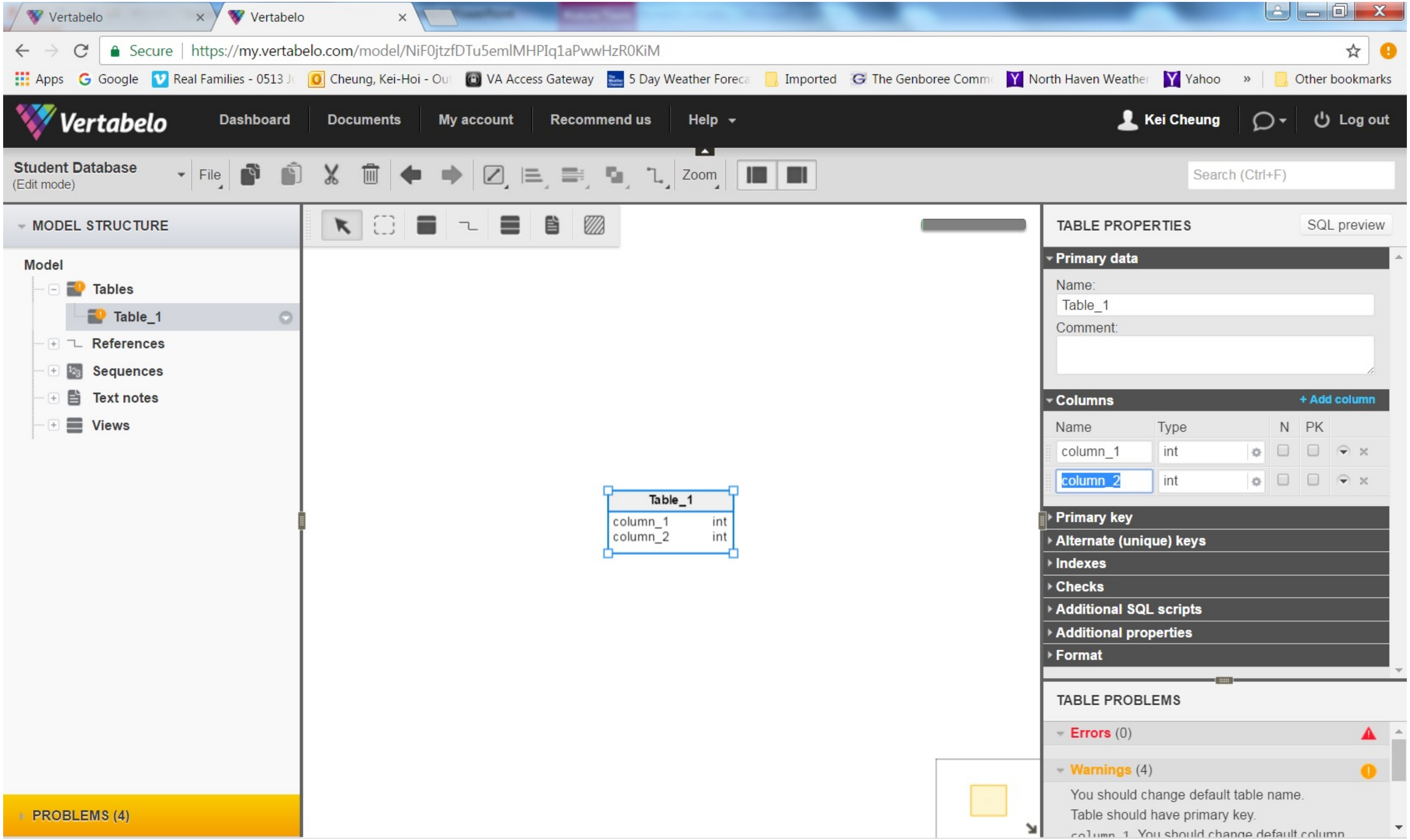
Model your career with Vertabelo!

We're looking for candidates for:

**Database Modeling Writer**
**(part-time remote freelance)**

with experience as an active professional database modeler, software or database architect – to write and publish original articles on Vertabelo's website.

Learn more »



Name:
Table_1

Comment:

| Name | Type | N | PK |
|---|---|---|---|
| column_1 | int | □ | □ |
| column_2 | int | □ | □ |

**Table_1**

| | |
|---|---|
| column_1 | int |
| column_2 | int |

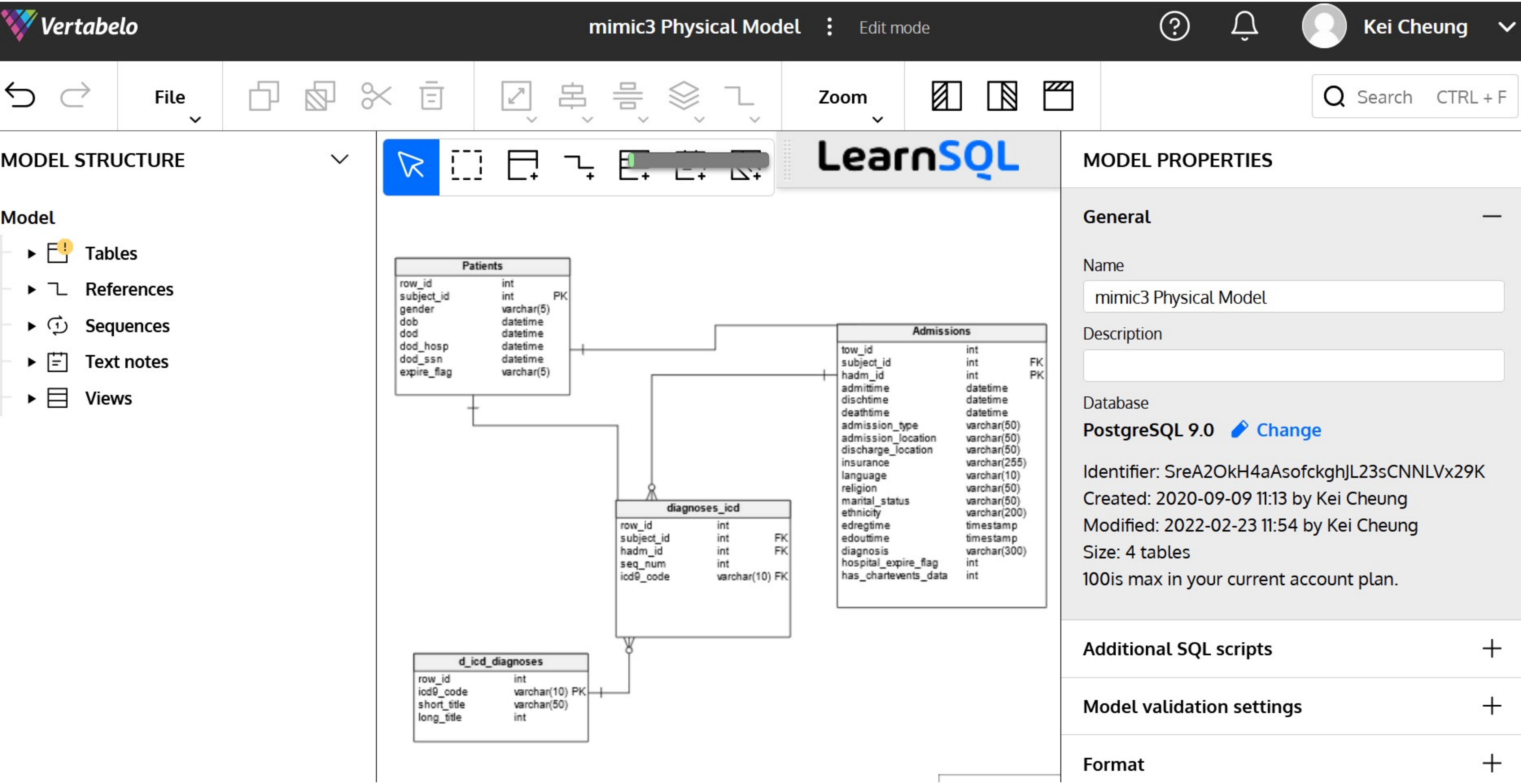
Errors (0)

Warnings (4)

You should change default table name.

Table should have primary key.

column_1 You should change default column



**Patients**

| | | |
|---|---|---|
| row_id | int | |
| subject_id | int | PK |
| gender | varchar(5) | |
| dob | datetime | |
| dod | datetime | |
| dod_hosp | datetime | |
| dod_ssn | datetime | |
| expire_flag | varchar(5) | |

**Admissions**

| | | |
|---|---|---|
| tow_id | int | |
| subject_id | int | FK |
| hadm_id | int | PK |
| admittime | datetime | |
| dischtime | datetime | |
| deathtime | datetime | |
| admission_type | varchar(50) | |
| admission_location | varchar(50) | |
| discharge_location | varchar(50) | |
| insurance | varchar(255) | |
| language | varchar(10) | |
| religion | varchar(50) | |
| marital_status | varchar(50) | |
| ethnicity | varchar(200) | |
| edregtime | timestamp | |
| edouttime | timestamp | |
| diagnosis | varchar(300) | |
| hospital_expire_flag | int | |
| has_chartevents_data | int | |

**diagnoses_icd**

| | | |
|---|---|---|
| row_id | int | |
| subject_id | int | FK |
| hadm_id | int | FK |
| seq_num | int | |
| icd9_code | varchar(10) | FK |

**d_icd_diagnoses**

| | | |
|---|---|---|
| row_id | int | |
| icd9_code | varchar(10) | PK |
| short_title | varchar(50) | |
| long_title | int | |

Name: mimic3 Physical Model

Description:

Database: PostgreSQL 9.0 Change

Identifier: SreA2OkH4aAsofckghJL23sCNNLVx29K

Created: 2020-09-09 11:13 by Kei Cheung

Modified: 2022-02-23 11:54 by Kei Cheung

Size: 4 tables

100is max in your current account plan.

# On-Line Transaction Processing (OLTP)

# What is OLTP?

- It is a class of information systems (e.g., databases) that facilitate and manage transaction-oriented applications, typically for data entry and retrieval transactions
- A database that is based on a normalized relational model is considered an OLTP application. It supports the following transactions:
  - Insert new rows
  - Update existing rows
  - Delete rows
  - Select rows
- A database transaction must be atomic, consistent, isolated and durable (ACID)

# Structured Query Language (SQL)

- It is a standard programming language for creating (CREATE) relational databases and tables as well as retrieving (SELECT), adding (INSERT), deleting (DELETE) and updating (UPDATE) data in a relational database
- It is compliant with ANSI and ISO standards

# SQL Statement (CREATE DATABASE/TABLE)

```
CREATE DATABASE Patient_DB;

CREATE TABLE Patient_DB.Patient
(
        ID int,
        Name varchar (50),
        Address varchar (250),
        Age smallint
        Sex varchar (2)
);
```

# INSERT Statement

```
INSERT INTO Patient_DB.Patient
(ID, Name, Address, Age, Sex)
VALUES (1, ‘John Doe’, ‘XYZ’, 40, ‘M’)
…
```

| ID | Name | Address | Age | Sex |
|---|---|---|---|---|
| 1 | John Doe | XYZ | 40 | M |
| 2 | Jane Smith | ABC | 34 | F |
| 3 | Mary Queen | PQSRT | 46 | F |
| 4 | Mike Lee | DWQER | 60 | M |

# UPDATE Statement

```
UPDATE Patient_DB.Patient
SET AGE=41
WHERE ID=1
```

| ID | Name | Address | Age | Sex |
|---|---|---|---|---|
| 1 | John Doe | XYZ | 41 | M |
| 2 | Jane Smith | ABC | 34 | F |
| 3 | Mary Queen | PQSRT | 46 | F |
| 4 | Mike Lee | DWQER | 60 | M |

# DELETE Statement

```
DELETE Patient_DB.Patient
WHERE Name=‘Mike Lee’
```

| ID | Name | Address | Age | Sex |
|---|---|---|---|---|
| 1 | John Doe | XYZ | 41 | M |
| 2 | Jane Smith | ABC | 34 | F |
| 3 | Mary Queen | PQSRT | 46 | F |

# SELECT Statement

```
SELECT ID, Name, Age, Sex
FROM Patient_DB.Patient
WHERE Age>=40
ORDER BY Age
```

| ID | Name | Address | Age | Sex |
|---|---|---|---|---|
| 1 | John Doe | XYZ | 40 | M |
| 2 | Jane Smith | ABC | 34 | F |
| 3 | Mary Queen | PQSRT | 46 | F |
| 4 | Mike Lee | DWQER | 60 | M |

# SELECT Statement (Aggregation)

```
SELECT Sex, avg(Age)
FROM Patient_DB.Patient
GROUP BY SEX
```

Results: M 50
F 40

| ID | Name | Address | Age | Sex |
|---|---|---|---|---|
| 1 | John Doe | XYZ | 40 | M |
| 2 | Jane Smith | ABC | 34 | F |
| 3 | Mary Queen | PQSRT | 46 | F |
| 4 | Mike Lee | DWQER | 60 | M |

# SELECT Statement (JOIN)

```
SELECT A.*, B.Report_Text
FROM Patient_DB.Patient AS A
INNER JOIN Patient_DB.LabTest. AS B
ON A.ID = B.Patient_ID
```

| ID | Name | Address | Age | Sex |
|---|---|---|---|---|
| 1 | John Doe | XYZ | 40 | M |
| 2 | Jane Smith | ABC | 34 | F |
| 3 | Mary Queen | PQSRT | 46 | F |
| 4 | Mike Lee | DWQER | 60 | M |

| Patient_ID | ID | Report_Text |
|---|---|---|
| 1 | 1 | …… |
| 1 | 2 | ……. |

# Other Types of SQL Statements

- TRUNCATE TABLE
- DROP TABLE
- CREATE VIEW
- CREATE INDEX (boost query performace)
  - Full-Text index (e.g., part of MS SQLServer)

# From OLTP to OLAP (On-Line Analytical Processing)

# OLAP Overview

- OLTP databases are tuned to small/medium size of data with relatively simple queries
- Some applications use fewer but more time-consuming analytic queries
- New architectures (data warehouses) have been developed to handle such analytic queries efficiently (De-normalization)

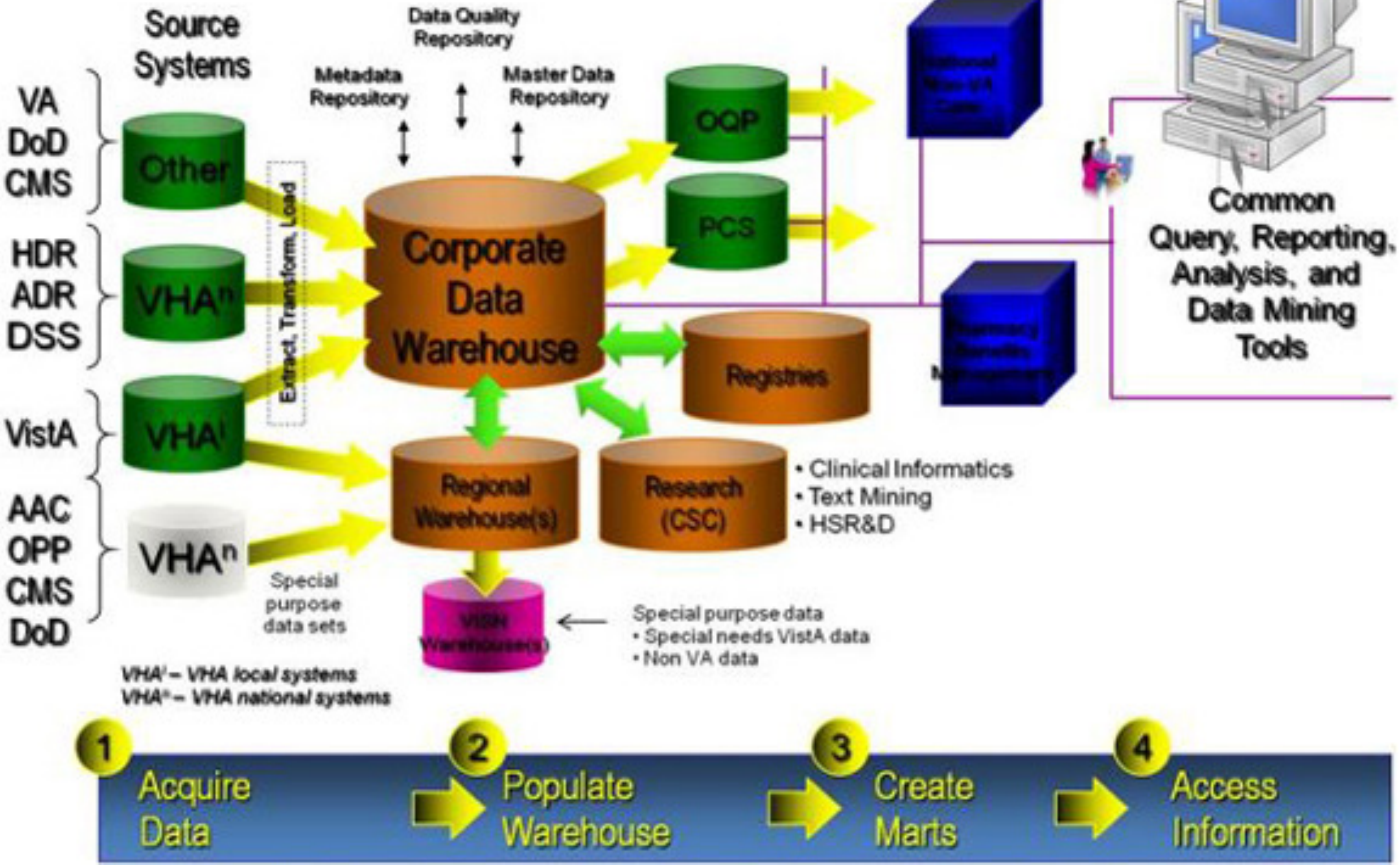
# OLAP Example Queries

- Amazon analyzes purchases by its customers to identify products of likely interest to customers
- Analysts at Wal-Mart look for merchandise items with increasing sales in some region

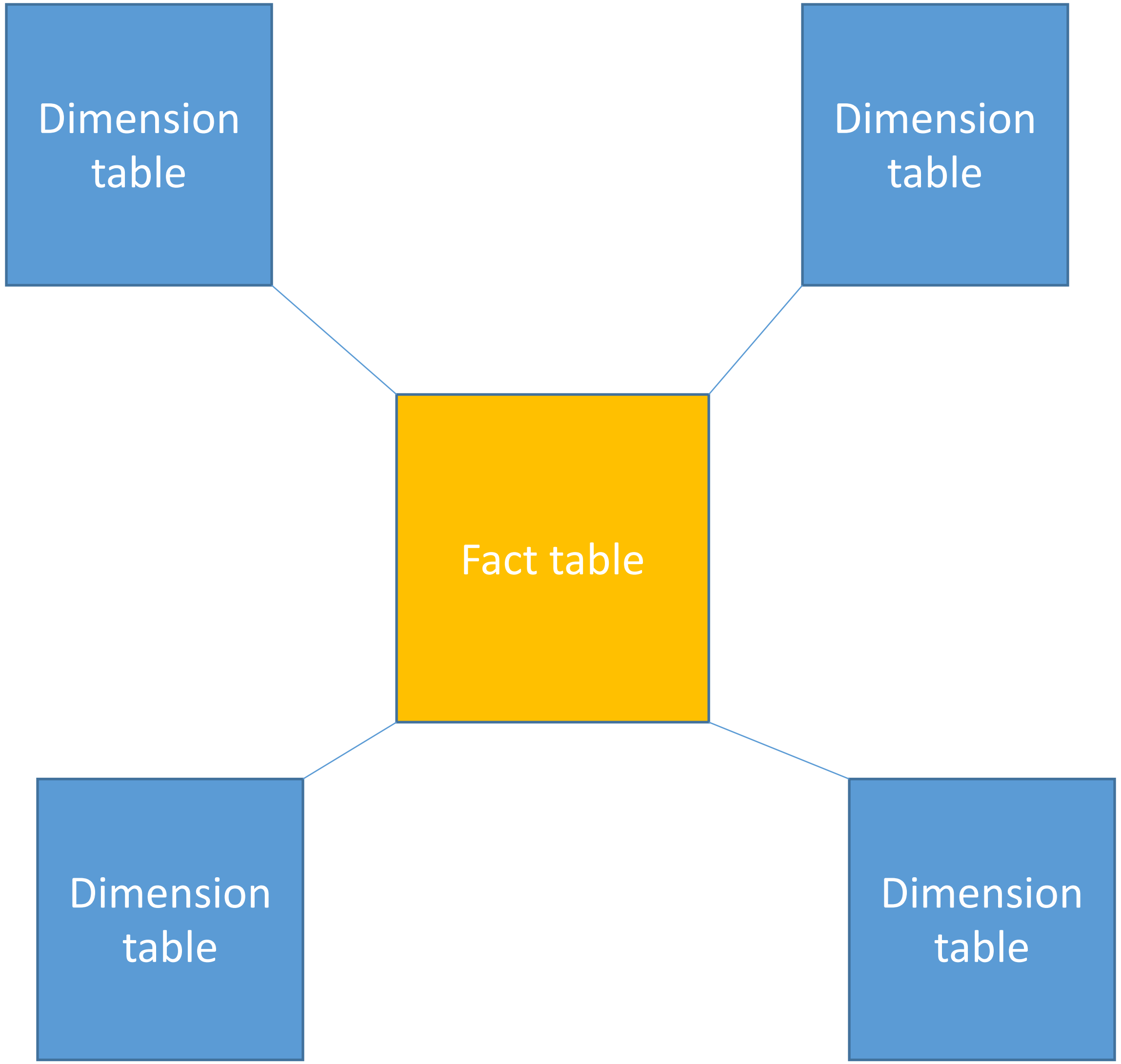
# Data Warehouse

- The most common form of database integration
  - Copy source databases into a single database (data warehouse)
  - Update the data warehouse periodically (in batch mode)
  - Support analytic queries using a dimensional data model (vs. a normalized entity-relationship model)
- Example: VA CDW

# VHA Data Warehousing Visual Architecture – Current Vision



# Star Schema



Dimension table

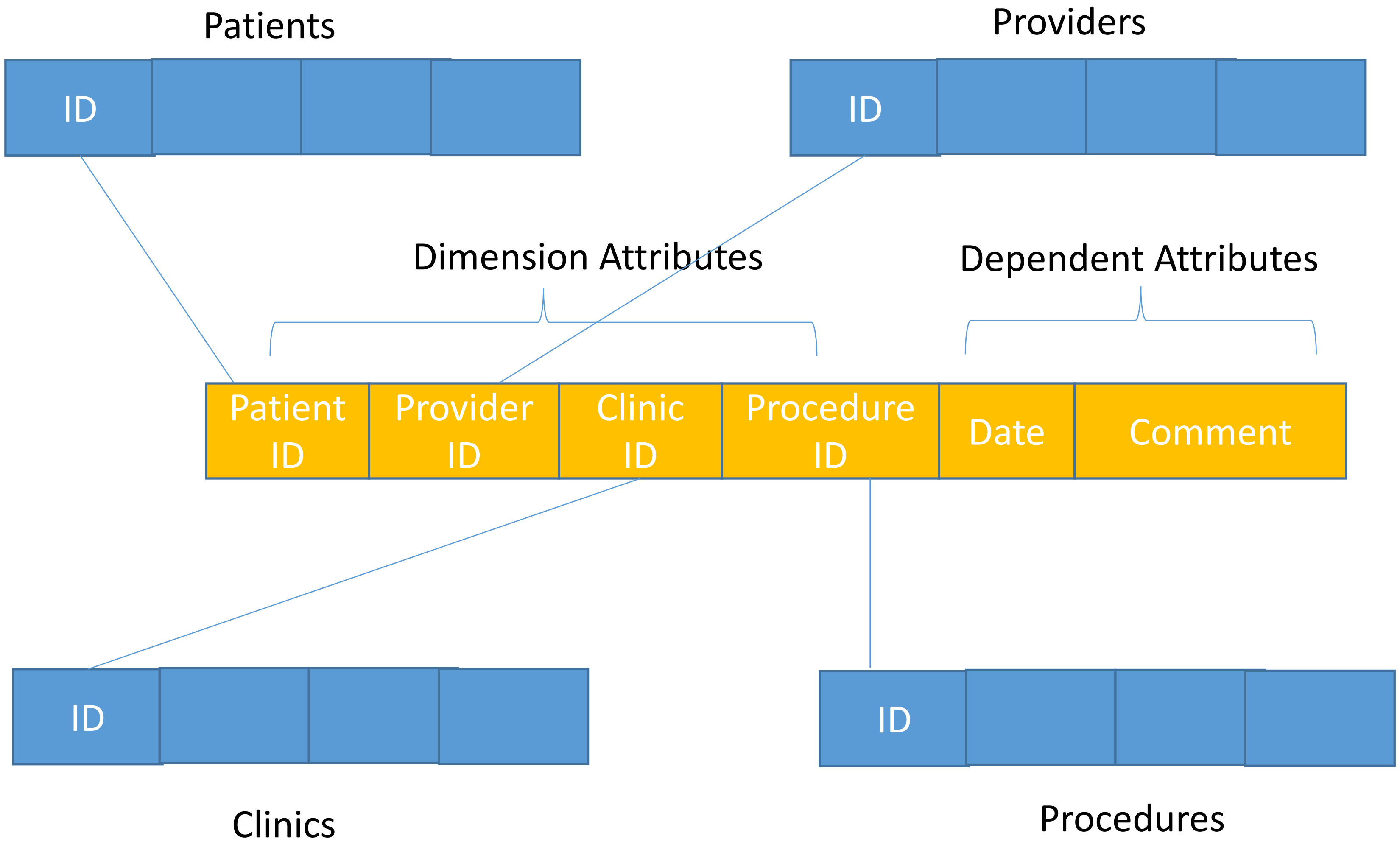
Dimension table

Fact table

Dimension table

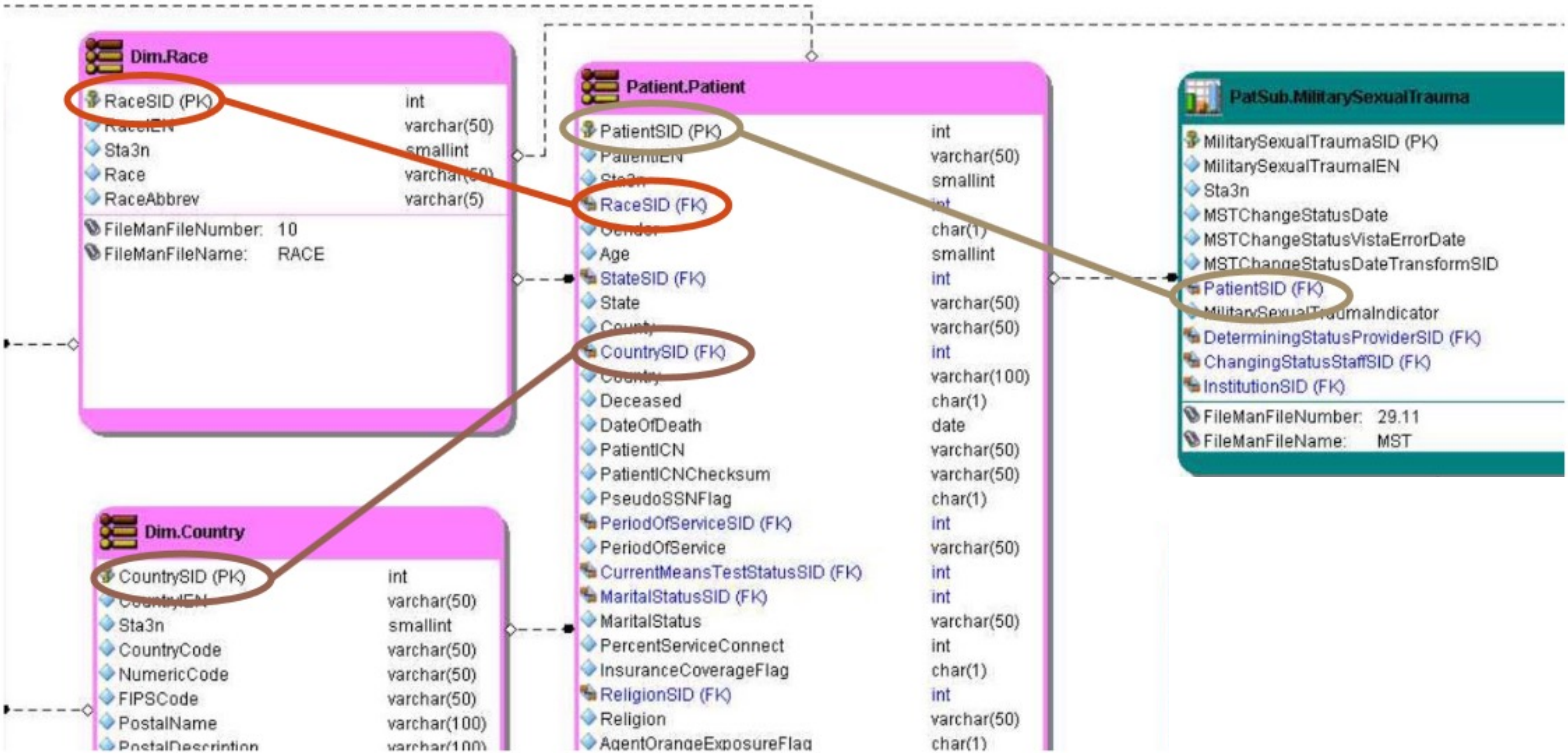
Dimension table

# Star Schema Example



# Example Queries

- Compare numbers of patient visits across different clinics for a given year
- Which are the top 10 most performed procedures among all clinics from 2010 to 2014



**Dim.Race**

| | |
|---|---|
| RaceSID (PK) | int |
| RaceIEN | varchar(50) |
| Sta3n | smallint |
| Race | varchar(50) |
| RaceAbbrev | varchar(5) |
| FileManFileNumber: | 10 |
| FileManFileName: | RACE |

**Patient.Patient**

| | |
|---|---|
| PatientSID (PK) | int |
| PatientIEN | varchar(50) |
| Sta3n | smallint |
| RaceSID (FK) | int |
| Gender | char(1) |
| Age | smallint |
| StateSID (FK) | int |
| State | varchar(50) |
| County | varchar(50) |
| CountrySID (FK) | int |
| Country | varchar(100) |
| Deceased | char(1) |
| DateOfDeath | date |
| PatientICN | varchar(50) |
| PatientICNChecksum | varchar(50) |
| PseudoSSNFlag | char(1) |
| PeriodOfServiceSID (FK) | int |
| PeriodOfService | varchar(50) |
| CurrentMeansTestStatusSID (FK) | int |
| MaritalStatusSID (FK) | int |
| MaritalStatus | varchar(50) |
| PercentServiceConnect | int |
| InsuranceCoverageFlag | char(1) |
| ReligionSID (FK) | int |
| Religion | varchar(50) |
| AgentOrangeExposureFlag | char(1) |

**PatSub.MilitarySexualTrauma**

| | |
|---|---|
| MilitarySexualTraumaSID (PK) | |
| MilitarySexualTraumaIEN | |
| Sta3n | |
| MSTChangeStatusDate | |
| MSTChangeStatusVistaErrorDate | |
| MSTChangeStatusDateTransformSID | |
| PatientSID (FK) | |
| MilitarySexualTraumaIndicator | |
| DeterminingStatusProviderSID (FK) | |
| ChangingStatusStaffSID (FK) | |
| InstitutionSID (FK) | |
| FileManFileNumber: | 29.11 |
| FileManFileName: | MST |

**Dim.Country**

| | |
|---|---|
| CountrySID (PK) | int |
| CountryIEN | varchar(50) |
| Sta3n | smallint |
| CountryCode | varchar(50) |
| NumericCode | varchar(50) |
| FIPSCode | varchar(50) |
| PostalName | varchar(100) |
| PostalDescription | varchar(100) |

# Beyond SQL

- NoSQL (graph databases like NEO4J, document databases like MongoDB)
- Semantic Web (standards for linked data and ontologies)

The End

Thanks!