

BITAH05 – Database technologies

Jasper Anckaert

Lecture 2 – Database schema, normalisation and MySQL Workbench

Previously

- Database
 - Collection of data that needs to be stored
 - Structured
 - Used everywhere
- Database system
 - Hardware – data – software – users
 - Storage space – quick – little redundancy – secure – clear structure
- Database Management System (DBMS)
 - Software application that interacts with the user, other applications, and the database itself to capture and analyse data
 - Storage – Retrieval – Manipulation – Authentication & authorization
- Relational databases – Relational Database Management System (RDBMS)
 - Enforce data integrity
 - Enforce referential integrity
 - Rules of E. Codd

Previously

- MySQL
 - Install, connect to and secure server
 - User – host – database – table
 - Privileges
 - Options file
- Create database
- Grant privileges
- Show databases, tables columns, create statement

Previously

- SQL
 - Data definition language
 - Statements to design database
 - CREATE, ALTER, DROP, ...
 - Data manipulation language
 - Statements to manage data
 - CRUD
 - SELECT, INSERT, UPDATE, DELETE
 - Data control language
 - Statements to manage database rights
 - GRANT, REVOKE

Previously

SQL: Structured Query Language

UPDATE clause { UPDATE movies Expression
SET clause { SET rating = rating + 1
WHERE clause { WHERE name = 'USA';

Expression
Predicate

} Statement

Previously

Column types

- INT
 - Integer
 - SIGNED: -2 147 483 648 tot 2 147 483 647
 - UNSIGNED: 0 tot 4 294 967 295
 - TINYINT, BIGINT, SMALLINT
- FLOAT & DOUBLE
 - Numbers with decimal point
 - FLOAT: 7 digits after decimal point, DOUBLE: 15 digits after decimal point
- DATE
 - YYYY-MM-DD
 - DATETIME
 - YYYY-MM-DD HH:MM:SS
 - ! TIMESTAMP ! No dates < 1970 and > 2038

Previously

Column types

- VARCHAR & CHAR
 - String with a certain number of characters
 - Define max number of characters e.g. VARCHAR(200)
 - VARCHAR: up to 65 535 characters
 - CHAR: up to 255 characters, spaces are added to reach required length

CHAR(10)



VARCHAR(10)



- VARCHAR is more efficient in storage, CHAR is faster for reading data
- Similar for INT vs BIGINT vs ...

Previously

Column types

- TEXT & BLOB
 - Used for texts that are not queried often or do not have to be searchable
 - BLOB for binary data (images, ...)
- ENUM
 - List of permitted values
 - E.g. Set of colours: 'red', 'green', 'blue'
 - Very efficient

Previously

Constraints

On top of column types, there are some additional requirements per column

- Primary key
 - Only 1 PK per table, all values must be unique
- UNIQUE
 - All values (or combinations) must be unique
- NOT NULL
 - Field can not be empty when adding data (empty = null)
- Default
 - Default value for a field
- Foreign key
 - Same constraints as referenced column
 - Security when adjusting linked data possible

Previously

- INSERT

```
INSERT INTO tbl (col1, col2) VALUES (val1, val2);
```

- SELECT

```
SELECT columns FROM tbl;
```

- ORDER BY

```
SELECT columns FROM tbl ORDER BY col1 [asc|desc] [, col2 [asc|desc]...];
```

- Calculated rows

- Built in functions for numbers, strings, dates

- Column aliases

- Can be used in the ORDER BY clause

- WHERE

```
SELECT columns FROM tbl WHERE condition(s) [ORDER BY sortcol];
```

- NULL values

```
SELECT ... WHERE col IS [NOT] NULL;
```

```
SELECT ifnull(col, value) ...
```

Previously

- AND, OR, NOT, XOR

- Boolean logic

- DISTINCT

`SELECT DISTINCT(cols) FROM ...`

- LIMIT, OFFSET

`SELECT ... LIMIT n [OFFSET x];`

- Aggregation

- Built in functions e.g. `count()`, `sum()`, `min()`, `max()`, ...

- GROUP BY

`SELECT [col,] aggregatefunctions FROM src [WHERE cond] GROUP BY col [ORDER BY ...];`

- HAVING

`SELECT [col,] aggregatefunctions FROM src [WHERE cond1] GROUP BY col HAVING cond2 [ORDER BY ...];`

Previously

Execution order

1. Input columns are determined
2. WHERE – input columns are filtered
3. GROUP BY – sorting & grouping of filtered input
4. Aggregation functions are calculated
5. HAVING – aggregation functions are filtered
6. ORDER BY – output is sorted
7. LIMIT/OFFSET – output is chopped

Previously

- JOIN

 - `SELECT * FROM tbl1 JOIN tbl2 ON tbl1.col1 = tbl2.col2;`

- INNER, LEFT, RIGHT, OUTER

- Foreign key

 - Primary key of other table
 - Index

- Relations between tables

 - 1:n one-to-many relationship
 - n:m many-to-many relationship (xref-table)

- Views

 - `CREATE VIEW viewname as SELECT ...`

- Index

 - 1 per query

Previously

- Allow redundancy

SNOWFLAKE	STAR
No redundancy	Redundant data
Easy to maintain and change	Less easy to maintain/change
Complex queries	Lower query complexity
Slower (more JOINS)	Faster
Uses less space	Uses more space (data is stored twice or more)
Bottom up	Top down

- DUMP

- Create database backup

Relational databases with MySQL - JOINS

Exercises (bioinf db)

- Give the accession number for the 3 longest human genes in the database
- How many genes are in the database for species with a genome size of at least 3000
- For the gene with accession number *NM_008220*, give
 - The length of the gene
 - The total genome size
- Retrieve all genes coming from a genome that was published in the first half of the year
- Retrieve all unique class names for model organisms with at least 10 chromosomes

MySQL Workbench

Database schema

- MySQL monitor to execute DDL commands
 - Servers
 - Advanced users
- GUI
 - HeidiSQL
 - MySQL Workbench

MySQL Workbench

Installation

- Available as for download (several operating systems)
 - <http://dev.mysql.com/downloads/workbench/>
- To install DEB package

```
# dpkg -i package.deb
```
- To install RPM package

```
# rpm -Uvh package.rpm
```
- To install on Windows/Mac
 - Double click package.msi or package.dmg

Welcome to MySQL Workbench

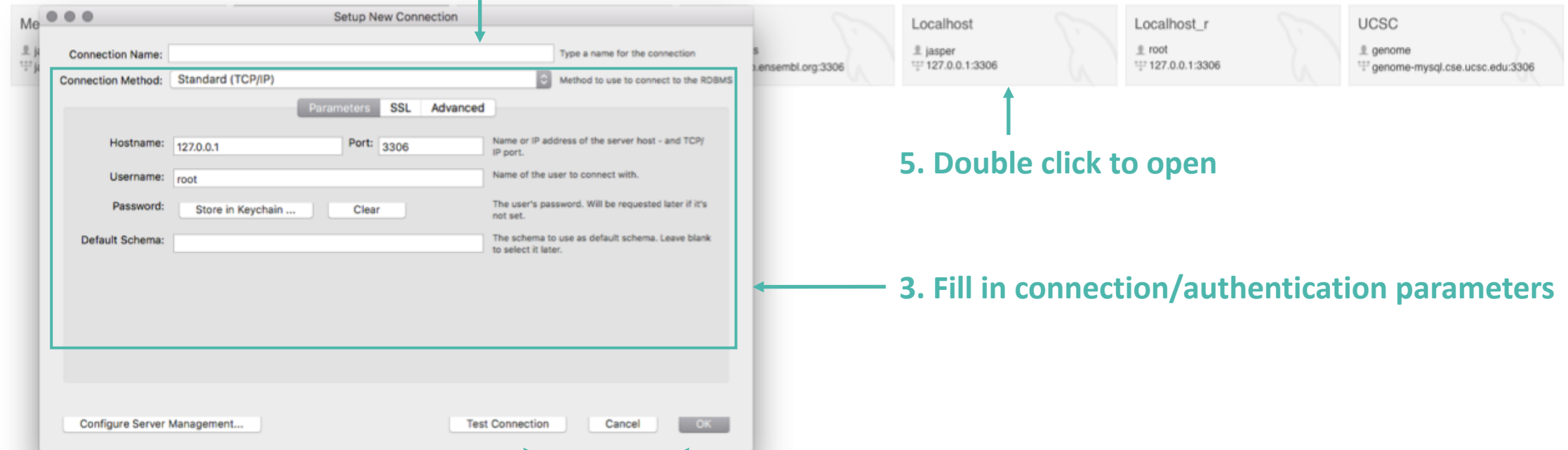
It allows you to design, create and browse your database schemas, work with database objects and insert data as well as design and run SQL queries to work with stored data. You can also migrate schemas and data from other database vendors to your MySQL database.

[Discuss on the Forums >](#)

1. Add connection (connection is saved and can be reused)

2. Choose a name for the database connection

MySQL Connections  



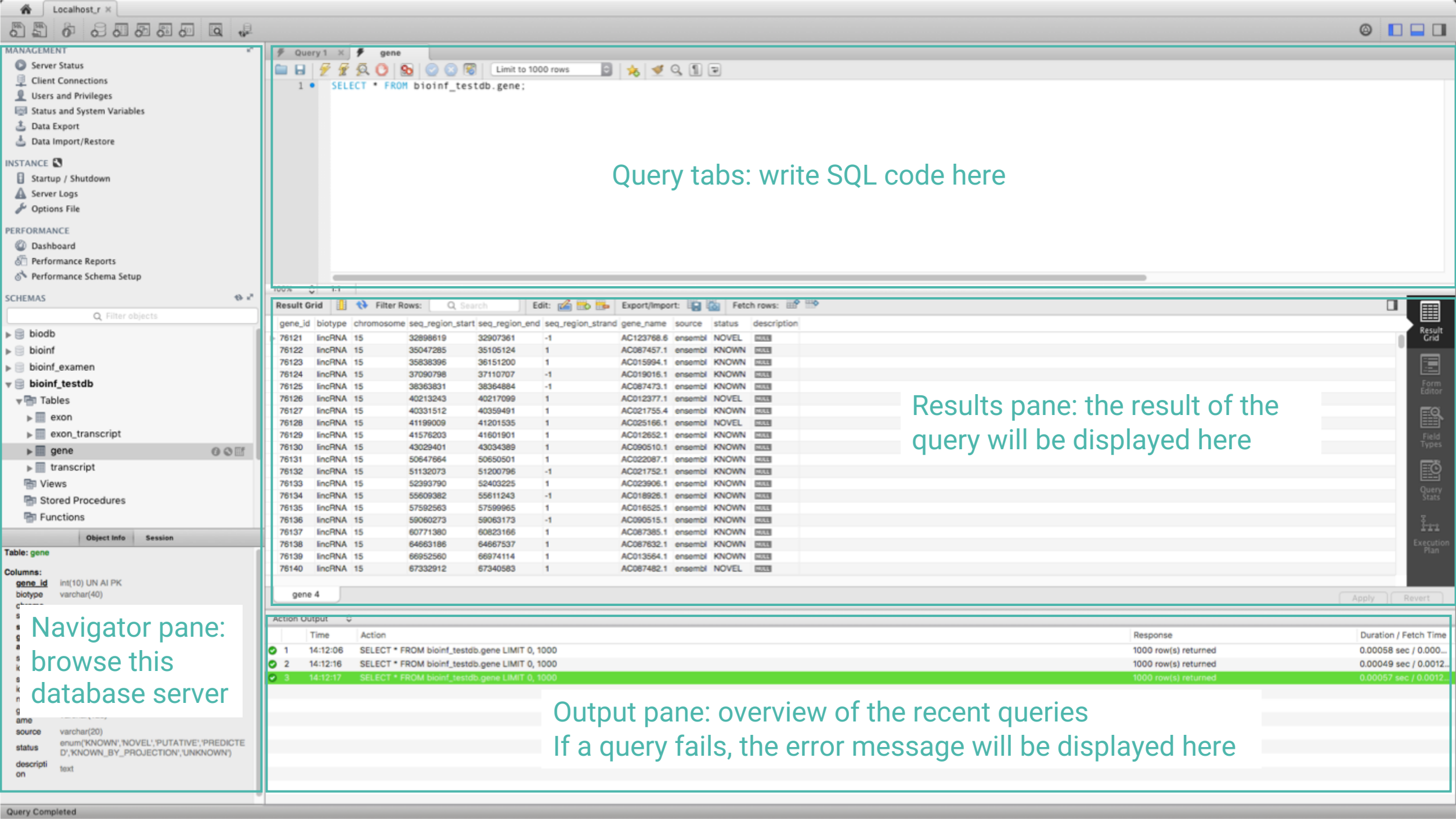
The screenshot shows the MySQL Workbench interface. In the foreground, the 'Setup New Connection' dialog box is open. It has a title bar with 'Me' and three window control buttons. The dialog is divided into several sections. At the top, there's a 'Connection Name' field with a placeholder text 'Type a name for the connection'. Below it is the 'Connection Method' dropdown, currently set to 'Standard (TCP/IP)', with a note 'Method to use to connect to the RDBMS'. There are three tabs: 'Parameters' (selected), 'SSL', and 'Advanced'. The 'Parameters' tab contains fields for 'Hostname' (127.0.0.1), 'Port' (3306), 'Username' (root), 'Password' (with 'Store in Keychain ...' and 'Clear' buttons), and 'Default Schema'. To the right of these fields are explanatory text blocks. At the bottom of the dialog are buttons for 'Configure Server Management...', 'Test Connection', 'Cancel', and 'OK'. In the background, the 'MySQL Connections' list is visible, showing three connections: 'Localhost' (jasper, 127.0.0.1:3306), 'Localhost_r' (root, 127.0.0.1:3306), and 'UCSC' (genome, genome-mysql.cse.ucsc.edu:3306). Each connection card has a MySQL logo icon.

Connection Name	Username	Host	Port
Localhost	jasper	127.0.0.1	3306
Localhost_r	root	127.0.0.1	3306
UCSC	genome	genome-mysql.cse.ucsc.edu	3306

5. Double click to open

3. Fill in connection/authentication parameters

4. Test and save the connection



MANAGEMENT

- Server Status
- Client Connections
- Users and Privileges
- Status and System Variables
- Data Export
- Data Import/Restore

INSTANCE

- Startup / Shutdown
- Server Logs
- Options File

PERFORMANCE

- Dashboard
- Performance Reports
- Performance Schema Setup

SCHEMAS

Filter objects

- biodb
- bioinf
- bioinf_examen
- bioinf_testdb
 - Tables
 - exon
 - exon_transcript
 - gene
 - transcript
 - Views
 - Stored Procedures
 - Functions

Table: gene

Columns:

- gene_id int(10) UN AI PK
- biotype varchar(40)
- c
- s
- g
- a
- k
- s
- k
- r
- g
- a
- m
- e
- s
- o
- u
- r
- c
- e
- s
- t
- a
- t
- u
- s
- d
- e
- s
- c
- r
- i
- p
- t
- i
- o
- n

source varchar(20)

status enum('KNOWN','NOVEL','PUTATIVE','PREDICTE D','KNOWN_BY_PROJECTION','UNKNOWN')

descripti on text

Navigator pane:
browse this
database server

Query 1 x gene

Limit to 1000 rows

1 • SELECT * FROM bioinf_testdb.gene;

Query tabs: write SQL code here

Result Grid

Filter Rows: Search Edit: Export/Import: Fetch rows:

gene_id	biotype	chromosome	seq_region_start	seq_region_end	seq_region_strand	gene_name	source	status	description
76121	lincRNA	15	32898619	32907361	-1	AC123768.6	ensembl	NOVEL	NULL
76122	lincRNA	15	35047285	35105124	1	AC087457.1	ensembl	KNOWN	NULL
76123	lincRNA	15	35838396	36151200	1	AC015994.1	ensembl	KNOWN	NULL
76124	lincRNA	15	37090798	37110707	-1	AC019016.1	ensembl	KNOWN	NULL
76125	lincRNA	15	38363831	38364884	-1	AC087473.1	ensembl	KNOWN	NULL
76126	lincRNA	15	40213243	40217099	1	AC012377.1	ensembl	NOVEL	NULL
76127	lincRNA	15	40331512	40358491	1	AC021755.4	ensembl	KNOWN	NULL
76128	lincRNA	15	41199009	41201535	1	AC025166.1	ensembl	NOVEL	NULL
76129	lincRNA	15	41576203	41601901	1	AC012652.1	ensembl	KNOWN	NULL
76130	lincRNA	15	43029401	43034389	1	AC090510.1	ensembl	KNOWN	NULL
76131	lincRNA	15	50647684	50650501	1	AC022087.1	ensembl	KNOWN	NULL
76132	lincRNA	15	51132073	51200796	-1	AC021752.1	ensembl	KNOWN	NULL
76133	lincRNA	15	52393790	52403225	1	AC023906.1	ensembl	KNOWN	NULL
76134	lincRNA	15	55609382	55611243	-1	AC018926.1	ensembl	KNOWN	NULL
76135	lincRNA	15	57592563	57599965	1	AC016525.1	ensembl	KNOWN	NULL
76136	lincRNA	15	59060273	59063173	-1	AC090515.1	ensembl	KNOWN	NULL
76137	lincRNA	15	60771380	60823166	1	AC087385.1	ensembl	KNOWN	NULL
76138	lincRNA	15	64663186	64667537	1	AC087632.1	ensembl	KNOWN	NULL
76139	lincRNA	15	66952560	66974114	1	AC013564.1	ensembl	KNOWN	NULL
76140	lincRNA	15	67332912	67340583	1	AC087482.1	ensembl	NOVEL	NULL

gene 4

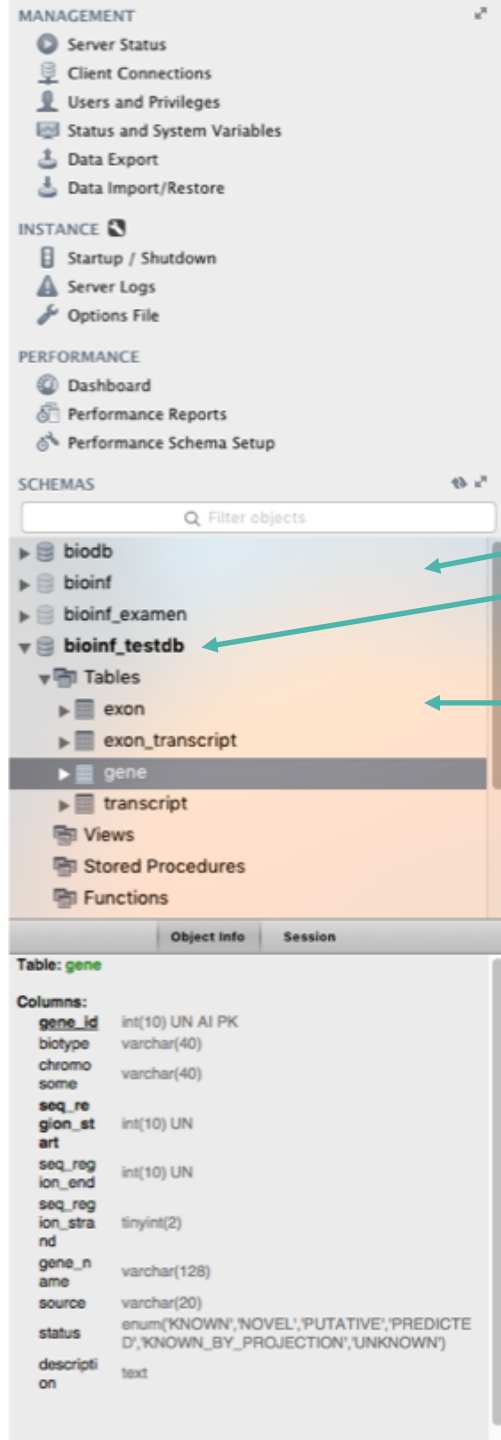
Apply Revert

Results pane: the result of the
query will be displayed here

Action Output

	Time	Action	Response	Duration / Fetch Time
✓ 1	14:12:06	SELECT * FROM bioinf_testdb.gene LIMIT 0, 1000	1000 row(s) returned	0.00058 sec / 0.000...
✓ 2	14:12:16	SELECT * FROM bioinf_testdb.gene LIMIT 0, 1000	1000 row(s) returned	0.00049 sec / 0.0012...
✓ 3	14:12:17	SELECT * FROM bioinf_testdb.gene LIMIT 0, 1000	1000 row(s) returned	0.00057 sec / 0.0012...

Output pane: overview of the recent queries
If a query fails, the error message will be displayed here



Different databases on this server

BOLD = currently active database, all queries will be executed in this db, double click to change

Browse tables in the db here

More information on the currently selected item

MySQL Workbench

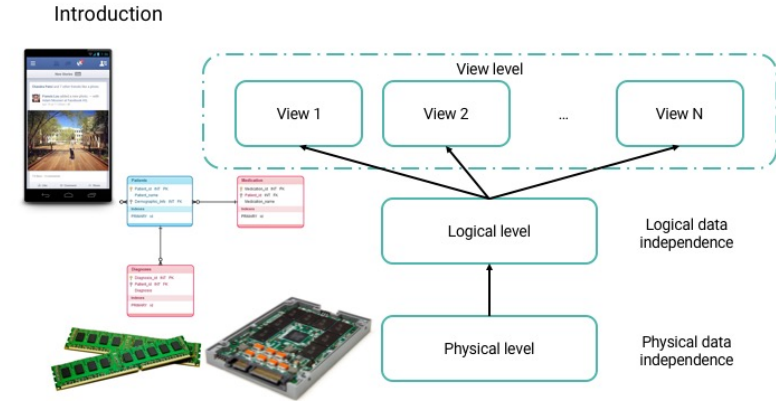
Exercices

- Connect to the MySQL database server
- Explore the server
 - How many databases are available to you?
 - How many tables does each database have?
 - What are the column types of the gene table (bioinf_testdb)?

MySQL Workbench

Data model

- Determines the structure of data
 - Conceptual data model
 - Structure of and relations between entities
 - Entity Relationship Diagram
 - Logical data model
 - Structure of and references between tables
 - Relations → foreign key constraints
 - Data Structure Diagram
 - Physical data model
 - Physical means by which data are stored (partitions, CPUs, tablespaces, ...)



MySQL Workbench

Database models

- Flat model
 - Single two-dimensional array of data elements
 - E.g. spreadsheet
- Hierarchical model
 - Data is organized into a tree-like structure
 - Records are connected through links
- Network model
 - Each record can have multiple parents and child records

MySQL Workbench

Database models

- Relational model
 - Tables are relations
 - Links between tables are not explicitly defined → use keys
 - What we've been using so far but with deviations
- Object-relational model
 - Relational model with object-oriented features
 - PostgreSQL
- Object oriented model
 - Data is represented in the form of objects
 - Use same model of representation as in programming language

MySQL Workbench

Creating a database - Normalisation

- Organizing columns and tables
 - Reduce redundancy
 - Improve integrity
- Remember E. Codd?

MySQL Workbench

Normalisation

- UNF
 - Unnormalized form
 - Group all data in one entity
- 1NF
 - Eliminate repeating (and calculated) groups in individual tables
 - Create separate table for each set of related data
 - Identify each set of related data with a primary key
- 2NF
 - Every non-prime attribute of the table is dependent on the whole key of every candidate key
- 3NF
 - Every non-prime attribute is non-transitively dependent on every key

MySQL Workbench

Normalisation

- BCNF
 - Any attribute on which some other attribute is fully functionally dependent = determinant
 - Every determinant is a candidate key
- 4NF – ETNF – 5NF – 6NF – DKNF

MySQL Workbench

Normalisation – example

Patient_no	Patient_name	Appointment_id	Time	Doctor
1	John	0	09:00	Zorro
2	Kerr	0	09:00	Killer
3	Adam	1	10:00	Zorro
4	Robert	0	13:00	Killer
5	Zane	1	14:00	Zorro

UNF DB(Patno,PatName,appNo,time,doctor)

1NF DB(Patno,PatName,appNo,time,doctor)

2NF DB(Patno,appNo,time,doctor)
R1(Patno,PatName)

3NF 2NF

BCNF DB(Patno,time,doctor)
R1(Patno,PatName)
R2(time,appNo)

MySQL Workbench

Normalisation – example

- Why is this table not in 1NF?
- Normalize up to 3NF
- Identify all keys in your 3NF relations

branchNo	branchAddress	telNos
B001	8 Jefferson Way, Portland, OR 97201	503-555-3618, 503-555-2727, 503-555-6534
B002	City Center Plaza, Seattle, WA 98122	206-555-6756, 206-555-8836
B003	14 – 8th Avenue, New York, NY 10012	212-371-3000
B004	16 – 14th Avenue, Seattle, WA 98128	206-555-3131, 206-555-4112

MySQL Workbench

Exercises

- Normalise up to 3NF (note: a procedure may occur on multiple dates)

Pet_id	Pet_name	Pet_type	Pet_age	owner	Visit_date	procedure
246	Rover	dog	12	Sam Cook	2002-01-13	01 – Rabies vaccination
					2005-03-27	10 - Examination
					2003-04-02	05 – Heart worm test
296	Spot	dog	2	Terry Kim	2002-01-21	08 – Tetanus vaccination
					200-03-10	05 – Heart worm test
341	Morris	cat	4	Sam Cook	2001-01-23	01– Rabies vaccination
					2002-01-13	01 – Rabies vaccination
519	Tweedy	bird	2	Terry Kim	2002-04-30	20 – Check up
					2002-04-30	12 – Eye wash

MySQL Workbench

Exercises

- Normalise up to 3NF

INVOICE

HILLTOP ANIMAL HOSPITAL
INVOICE # 987

DATE: JAN 13/2002

MR. RICHARD COOK
123 THIS STREET
MY CITY, ONTARIO
Z5Z 6G6

<u>PET</u>	<u>PROCEDURE</u>	<u>AMOUNT</u>
ROVER	RABIES VACCINATION	30.00
MORRIS	RABIES VACCINATION	24.00
TOTAL		54.00
TAX (8%)		<u>4.32</u>
AMOUNT OWING		<u>58.32</u>

MySQL Workbench

Exercises

- Normalise up to BCNF
 - Grade_report(StudNo,StudName,(Major,Adviser,(CourseNo,Ctitle,InstrucName,InstructLocn,Grade)))
 - Functional dependencies
 - StudNo -> StudName
 - CourseNo -> Ctitle,InstrucName
 - InstrucName -> InstructLocn
 - StudNo,CourseNo,Major -> Grade
 - StudNo,Major -> Advisor
 - Advisor -> Major

MySQL Workbench

Exercises

video(title,director,serial)
customer(name,addr,memberno)
hire(memberno,serial,date)

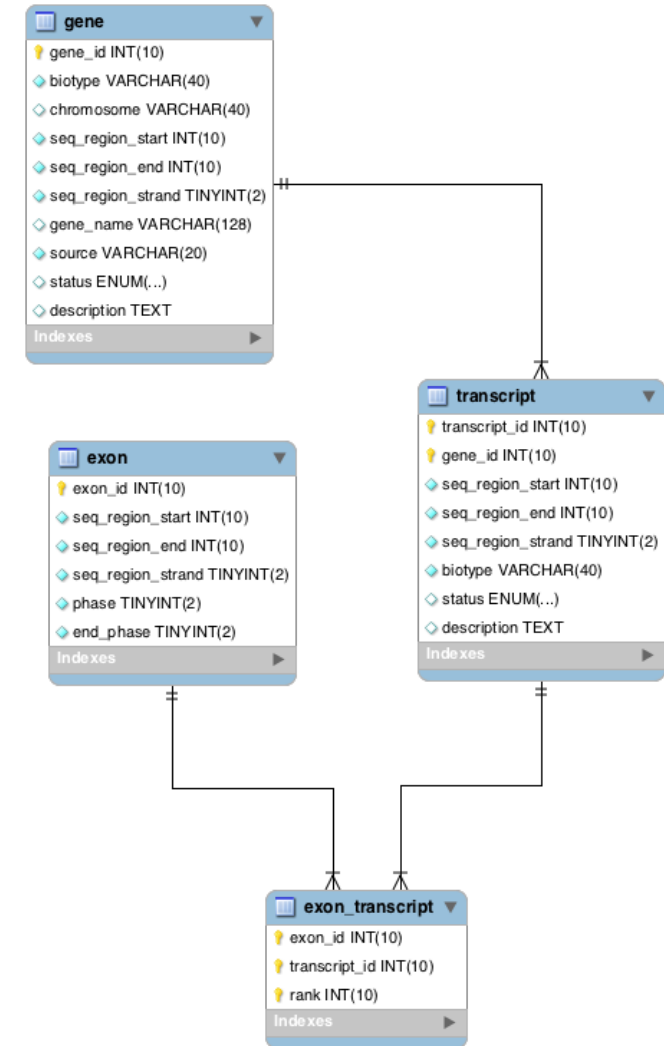
title->director,serial
serial->title serial->director
name,addr -> memberno
memberno -> name,addr
serial,date -> memberno

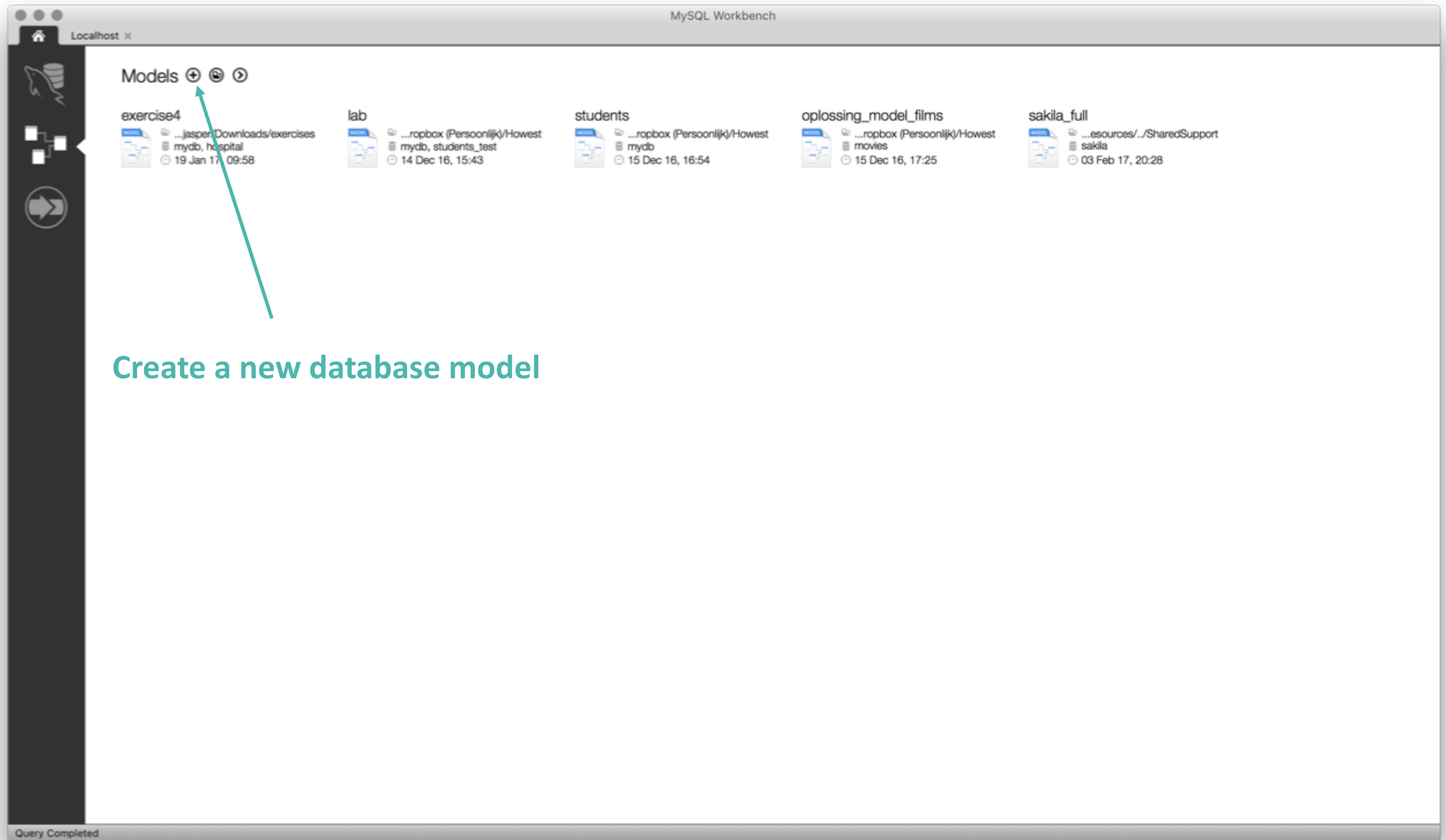
- What normal form is this?
- Convert to BCNF

MySQL Workbench

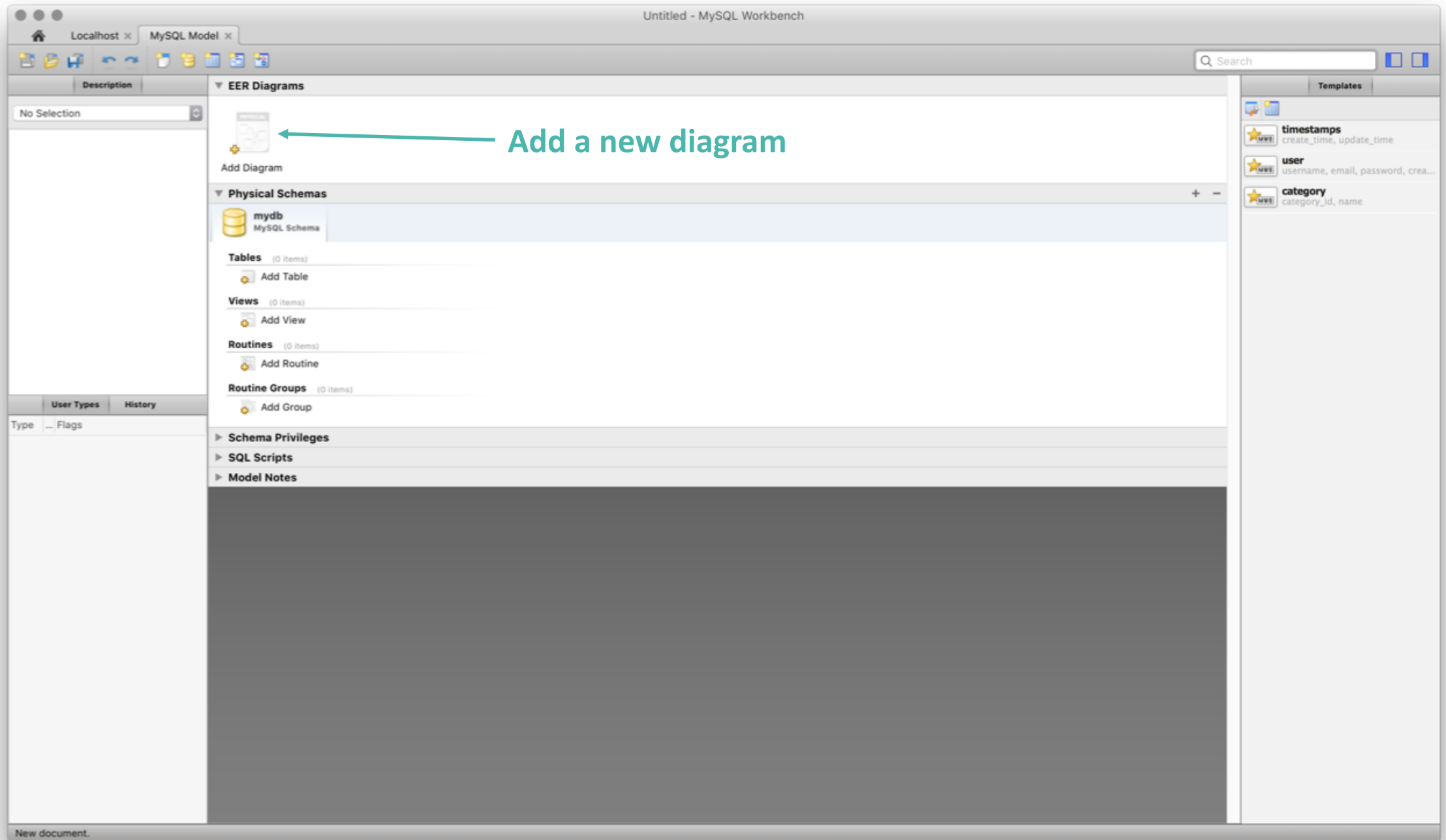
Creating tables

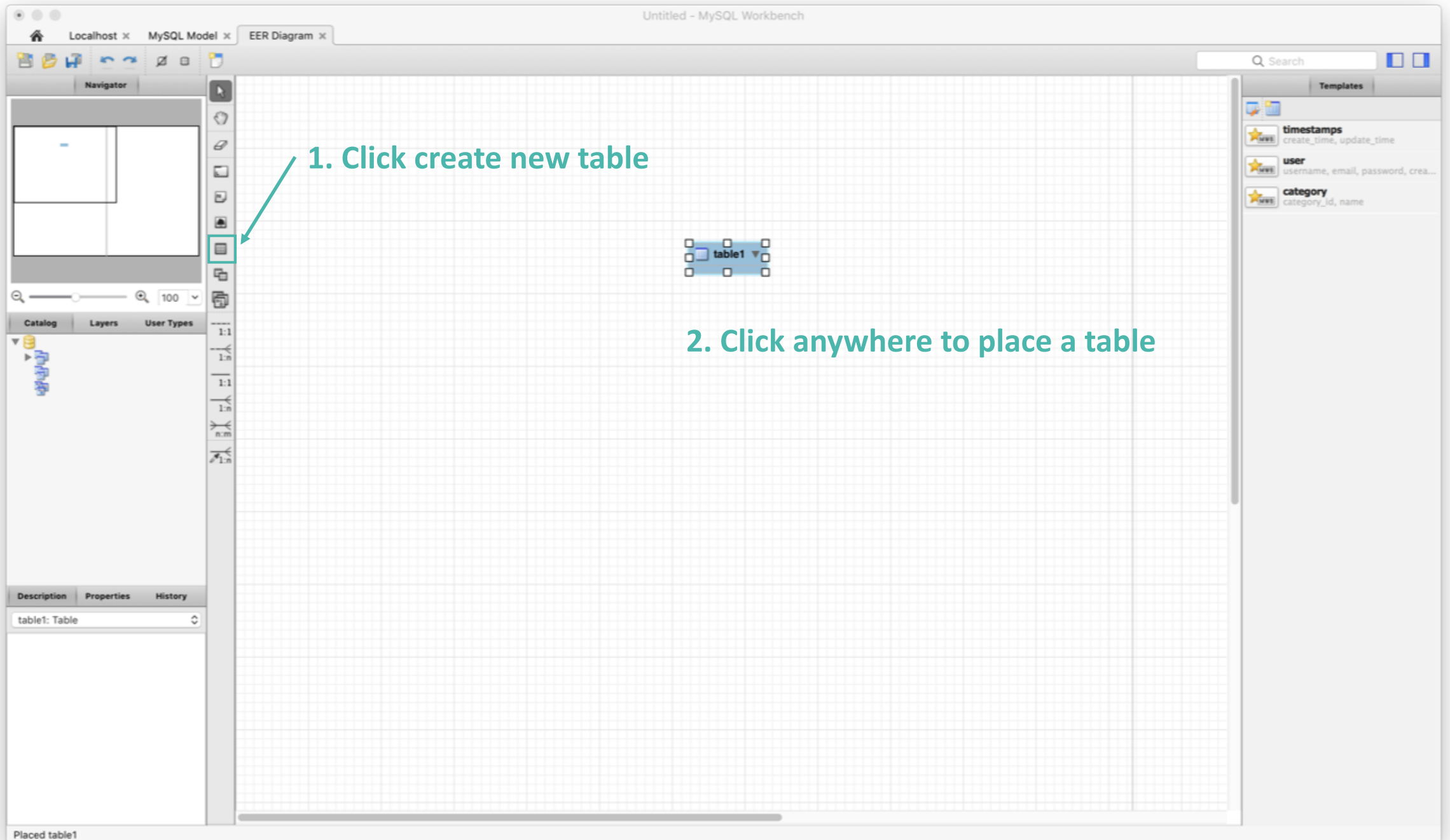
- Use the "model" interface in MySQL Workbench
 - DDL statements will be auto-generated
- Tables can be placed anywhere and dragged around
- Foreign keys will be displayed as lines and arrows





Create a new database model





Creating tables



2. Choose table name

3. Add columns

MySQL Workbench

Add columns to table

Column name

Primary key

UNIQUE

Auto increment

Column	Datatype	PK	NN	UQ	BIN	UN	ZF	AI	G	Default / Expression
ID	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
firstname	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
lastname	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
dateofbirth	DATE	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<click to edit>		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Columns Indexes Foreign Keys Triggers Partitioning Options Inserts Privileges

Column type

Not NULL

Unsigned (for numeric types e.g. INT)

MySQL Workbench

Add foreign keys to table

1. Make sure the table that holds the FK is selected

2. Switch to the FK tab

3. Add a new FK and select the referenced table

Column on THIS table

Column on REFERENCED table (PK)

Foreign Key: student_study, Referenced Table: 'mydb'. 'study'

Column	Referenced Column
<input type="checkbox"/> ID	
<input type="checkbox"/> firstname	
<input type="checkbox"/> lastname	
<input type="checkbox"/> dateofbirth	
<input checked="" type="checkbox"/> study_id	ID

On Update: NO ACTION
On Delete: NO ACTION
Comment:
☐ Skip on SQL generation

MySQL Workbench

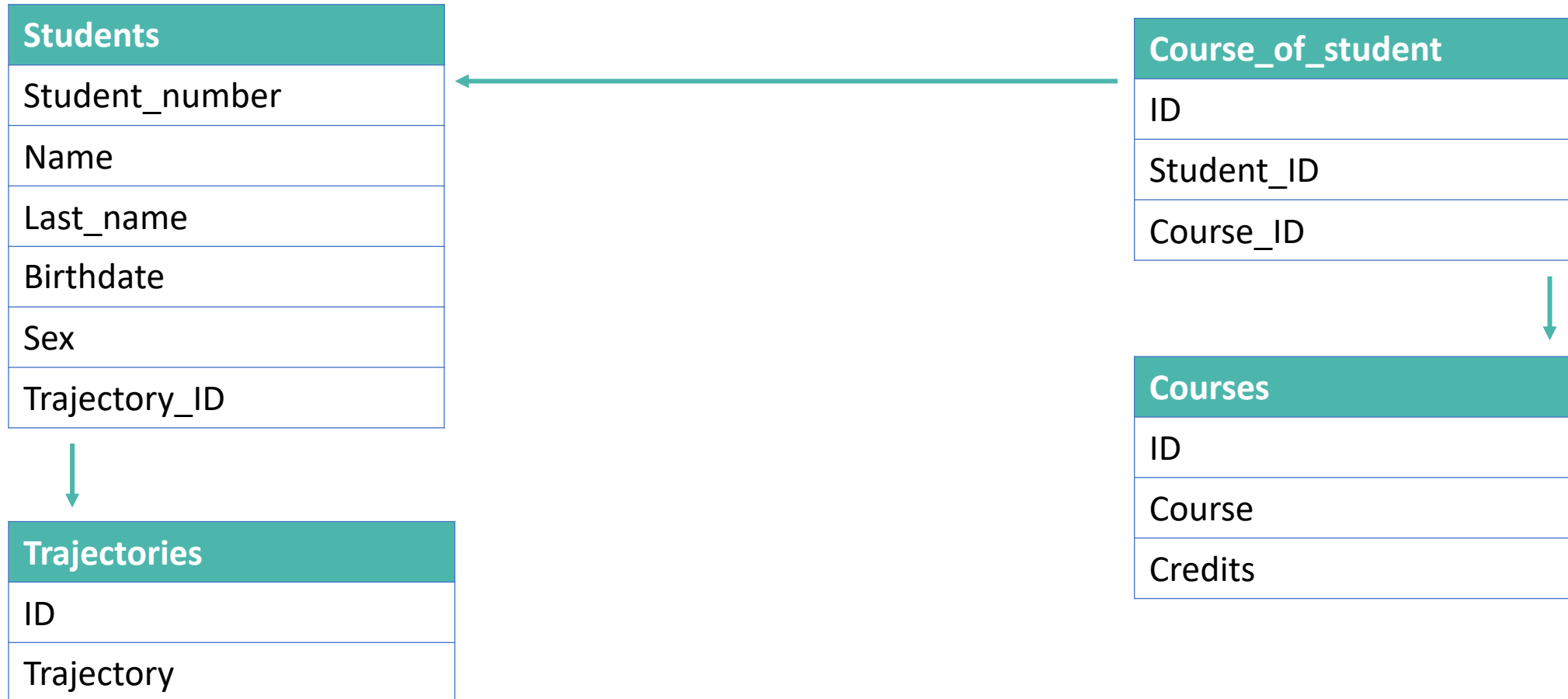
Excercises

- Export the data from the `modorg` table in your `biodb` database
- Empty your table (TRUNCATE)
- Import data into the `modorg` table using your export file

MySQL Workbench

Exercises

- Create the following database schema

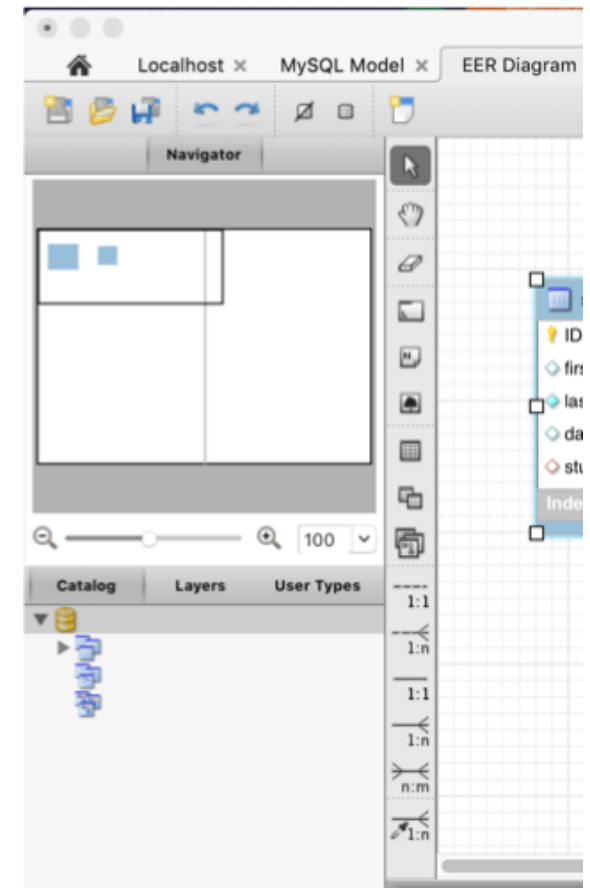


MySQL Workbench

Forward engineering

- Function in MySQL Workbench
 - Generates SQL code to create/modify a database based on your model
- Make sure the name of your database is correct!
 - Located under *Database > Forward engineer*
 - Check in the database browser (Refresh)

Double click to change →



MySQL Workbench

Exercises

- Create a MySQL table to track the movies you have watched:
 - Movie title
 - Genre: action, comedy, drama, horror, science fiction
 - Date you watched to movie
 - Score: 0-10
 - Comments
- Create a table to store your favourite directors and link it with the movie table
- Create a table to store your favourite actors and link it with the movie table
- Forward engineer your tables to your database
- Add some rows to the table you have created

MySQL Workbench

Creating a database

- Important questions
 - Which data?
 - Constraints?
 - Application?
 - Relations between data?



Entity relationship diagram

MySQL Workbench

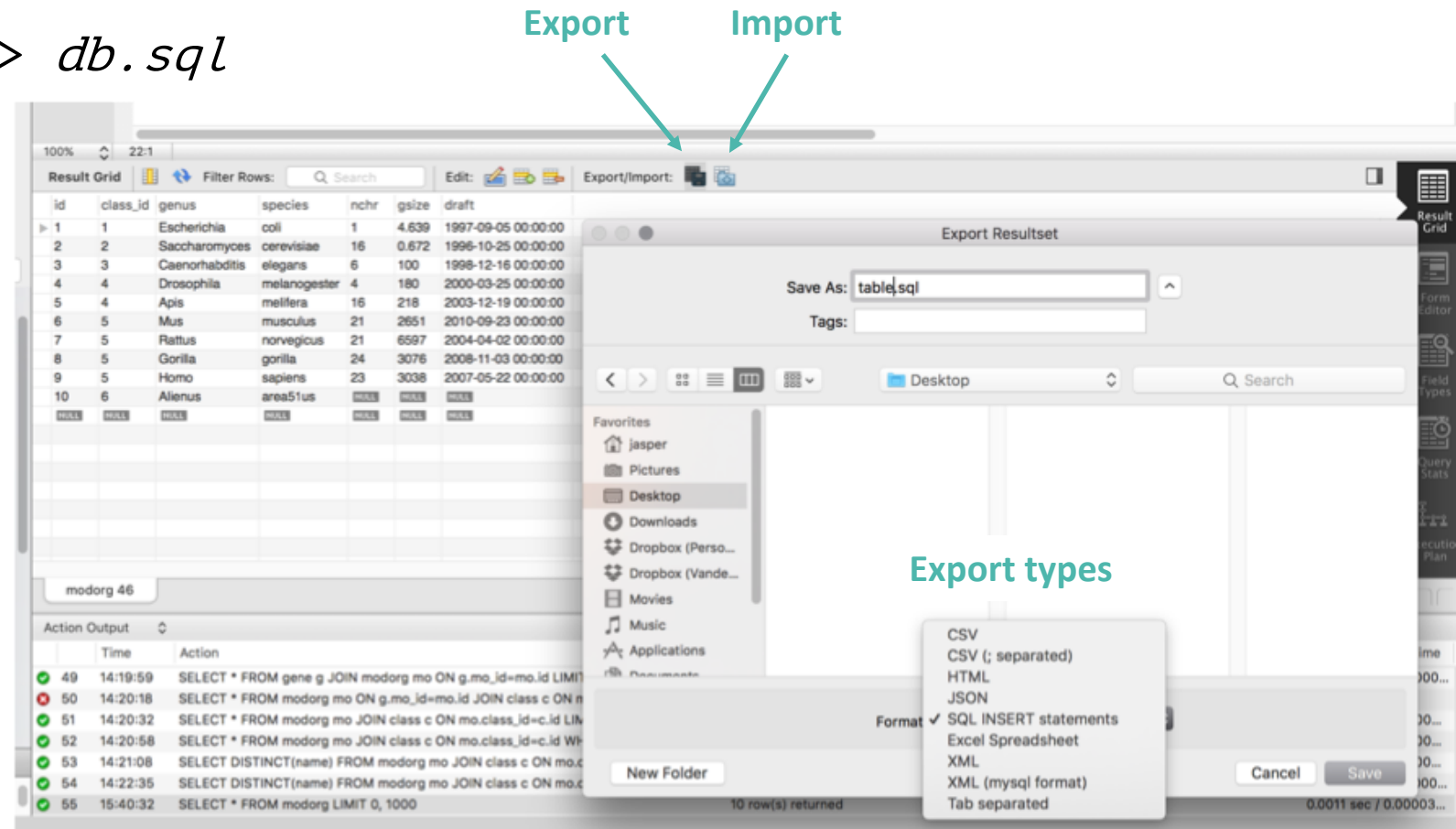
Exercises

- Reverse engineer the model of the bioinf_testdb
 - Check out the relationships between the different tables
 - Give the names of all the keys used

MySQL Workbench

Exporting data

- Remember
 - `$ mysqldump [opt] db > db.sql`
- Dump your database
 - Structure, data or both
 - Useful for backup
- Ability to export part of results (JSON, CSV, HTML, XML, ...)



MySQL Workbench

Import data

- Import entire dump file
- Import data from file (CSV, existing table, SQL, JSON)

