

Engineering Databases

Lecture 2 - SQL

October 26, 2022

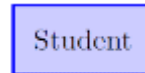
M. Saeed Mafipour & Mansour Mehranfar

Terms of lecture 1

- A **Database** is where all the data is stored
- A **Database Management System** is the administrative software of the DB
- A **schema** defines the structure of the data
- An **instance** is one set of data that complies to the schema
- A **data model** describes how we can model the schema in the database
- Our data model of choice is the **relational model**

- We use the **Entity Relationship Model** to design a schema

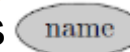
- **Entities** are well-defined physical objects or mental concepts



- **Relationship** are given between entity types



- **Attributes** characterize entities and relationships



- The **primary key** is an attribute (set) that identifies one entity

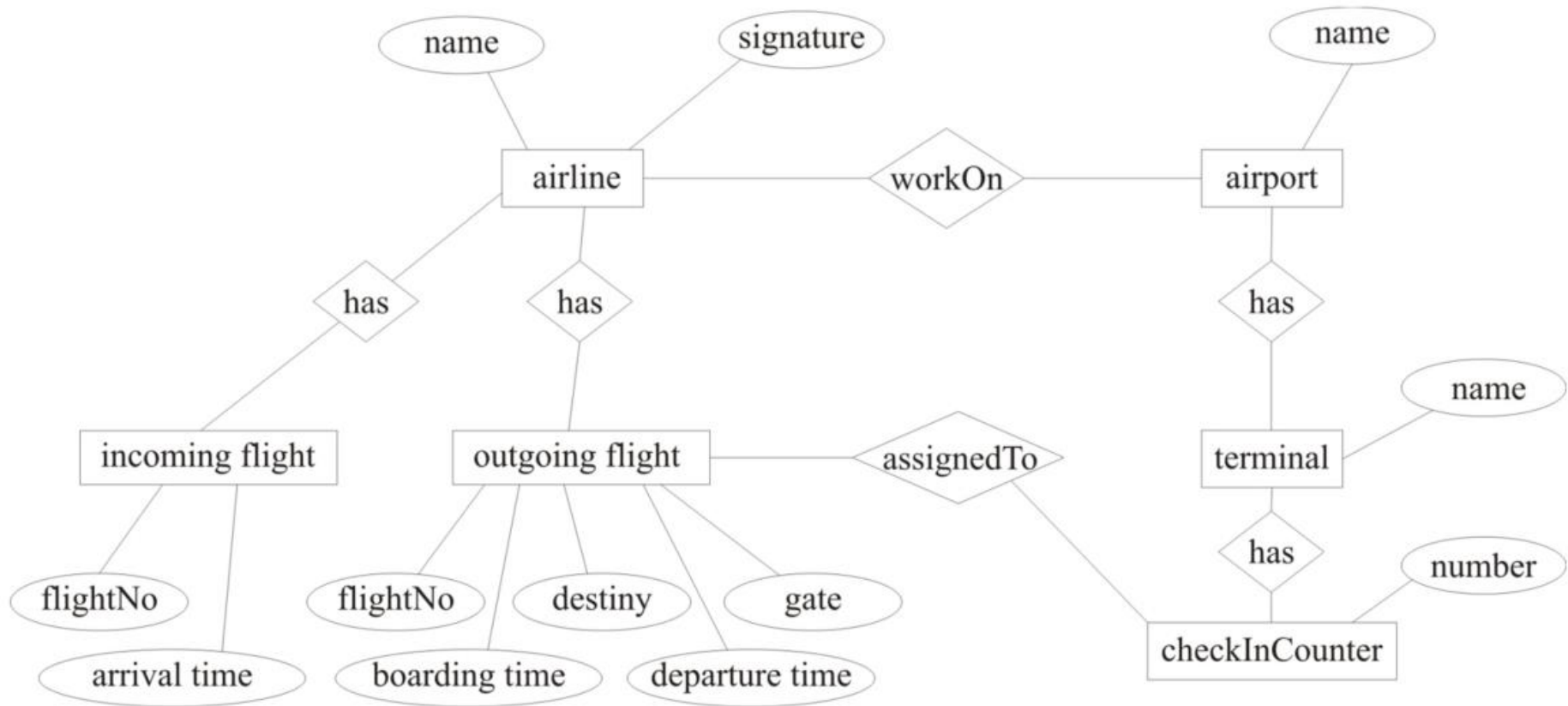


- **Cardinalities** represent consistency rules (Chen Notation, 1:1, 1:N, M:N)

Exercise: Entity-Relationship-Diagram

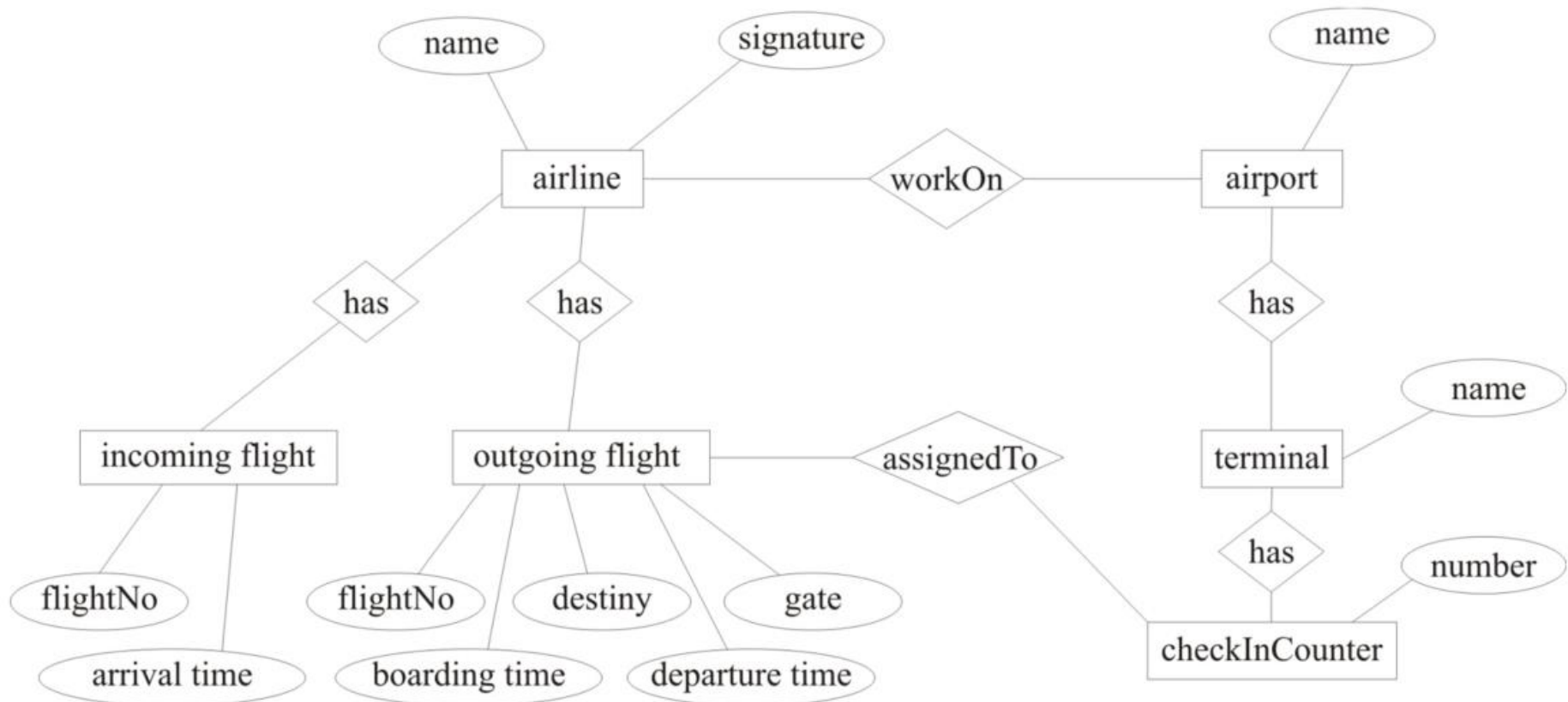
- Your task is to define the database structure for an airport. To capture the “universe-of-discourse” you use the entity-relationship-model. From the interviews with your clients you know the following:
- A number of airline companies work at the airport. The airline companies have a name and a signature (two letters) to identify their flights. Each airline company has a number of outgoing and incoming flights. Each outgoing flight has a flight number, a destiny, a gate, a boarding time and a departure time. Each incoming flight has a flight number, an origin, and an arrival time. The airport has different terminals. Each terminal has a name and a number of check-in counters. Each check-in-counter has a number. For each outgoing flight there is a specific check-in counter.
- Draw the entity-relationship-diagram.

Airport example: Solution



Airport example: Solution

- Add cardinalities using the Chen Notation (1:1, 1:N, M:N)



Relational Query Language SQL

- “Structured Query Language”
- Based on formal relational algebra
- Two sublanguages:
 - Data Definition Language
 - Define and modify schema
 - Data Manipulation Language
 - Add and remove data
 - Queries data
- SQL a declarative language
 - User defines what to find or to do
 - Database system decides how the query is processed

Data definition language

- Comprises methods to change the data structures
- CREATE TABLE statement
- DROP statement
- ALTER statement
- RENAME statement

Schema definition with SQL – CREATE TABLE

- The CREATE TABLE statement adds new tables to your database
- The syntax (grammar) is as follows:

```
CREATE TABLE <name of table>
(
    <name of attribute> <type of attribute> <integrity constraints>,
    <name of attribute> <type of attribute> <integrity constraints>, ...
);
```

- Example:

```
CREATE TABLE Courses (
    CourseNo INTEGER,
    Name VARCHAR(10),
    Room CHARACTER(5)
);
```



<placeholder>

SQL basic data types

- Each attribute (column) has a data type
- Numbers:
 - numeric/number: generic type
 - integer
 - float
 - date/time, datetime, e.g. date: 0001-01-01. . . 9999-12-31
 - time: 00:00:00.0000-23:59:59.9999
- Strings:
 - char(n)
 - varchar(n)
- Binary Data

SQL string data types

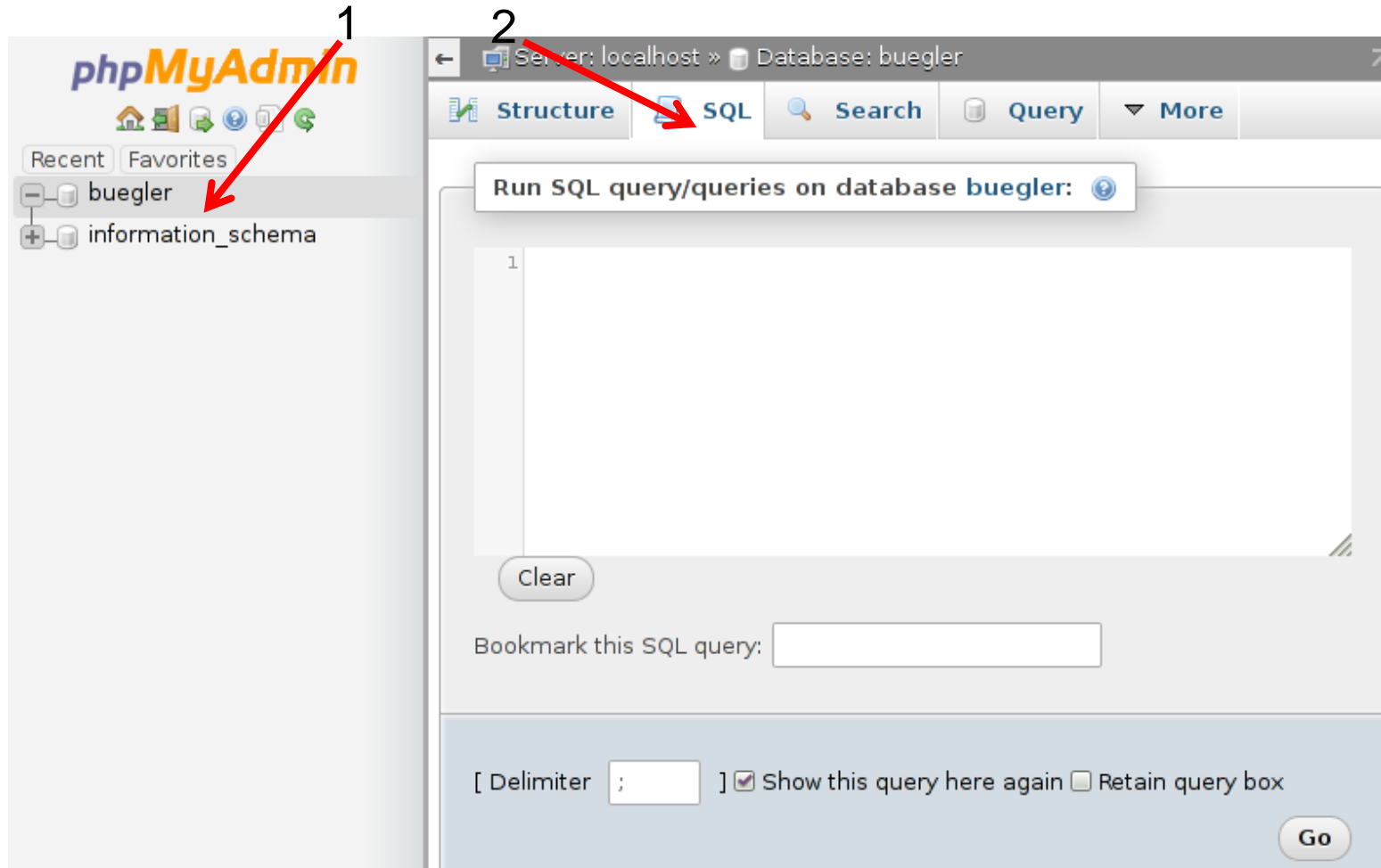
String types:

Data type	Description	Storage
char(n)	Fixed width character string. Maximum 8,000 characters	Defined width
varchar(n)	Variable width character string. Maximum 8,000 characters	2 bytes + number of chars
varchar(max)	Variable width character string. Maximum 1,073,741,824 characters	2 bytes + number of chars
text	Variable width character string. Maximum 2GB of text data	4 bytes + number of chars
nchar	Fixed width Unicode string. Maximum 4,000 characters	Defined width x 2
nvarchar	Variable width Unicode string. Maximum 4,000 characters	
nvarchar(max)	Variable width Unicode string. Maximum 536,870,912 characters	
ntext	Variable width Unicode string. Maximum 2GB of text data	
bit	Allows 0, 1, or NULL	
binary(n)	Fixed width binary string. Maximum 8,000 bytes	
varbinary	Variable width binary string. Maximum 8,000 bytes	
varbinary(max)	Variable width binary string. Maximum 2GB	
image	Variable width binary string. Maximum 2GB	

Integrity Constraints

- NOT NULL (data cannot be empty)
- UNIQUE (cannot be null, unique value in the table)
- PRIMARY KEY (identifier of the entity and unique)
- FOREIGN KEY (identifier of an entity of another table)
- DEFAULT (default value of a column)
- AUTO_INCREMENT (automatically set to 'last value + 1')
- Combinations possible
CREATE TABLE Lectures (
CourseNo INTEGER PRIMARY KEY AUTO_INCREMENT
);

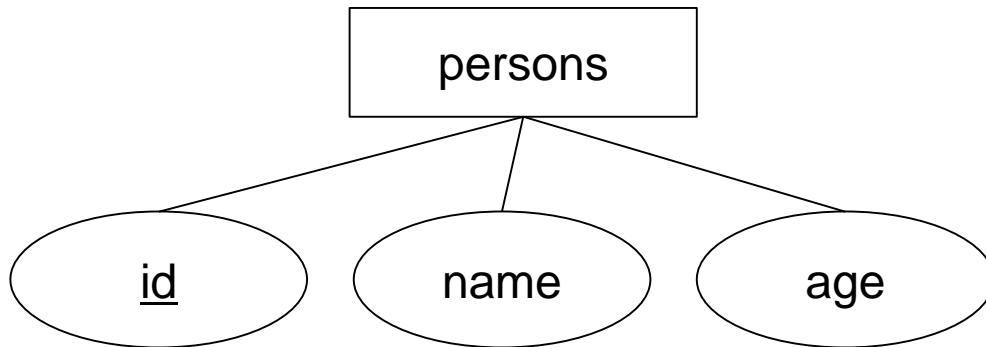
SQL Exercise 1



The screenshot shows the phpMyAdmin interface. On the left sidebar, the 'buegler' database is selected, indicated by a red arrow labeled '1'. The main panel shows the 'SQL' tab selected, indicated by a red arrow labeled '2'. The 'Run SQL query/queries on database buegler:' form is visible, with a text area containing the number '1'. Below the text area is a 'Clear' button. At the bottom, there is a 'Bookmark this SQL query:' input field, a 'Delimiter' dropdown set to ';', a checkbox for 'Show this query here again' which is checked, and a 'Retain query box' checkbox which is unchecked. A 'Go' button is at the bottom right.

SQL Exercise 1

- Create a table named persons having the columns “id”, “name”, and “age” using the CREATE TABLE statement
- Define a primary key for the id
- Use appropriate data types for the attributes



Change schema with SQL – ALTER TABLE

- Changes the structure of a table
- The syntax is as follows

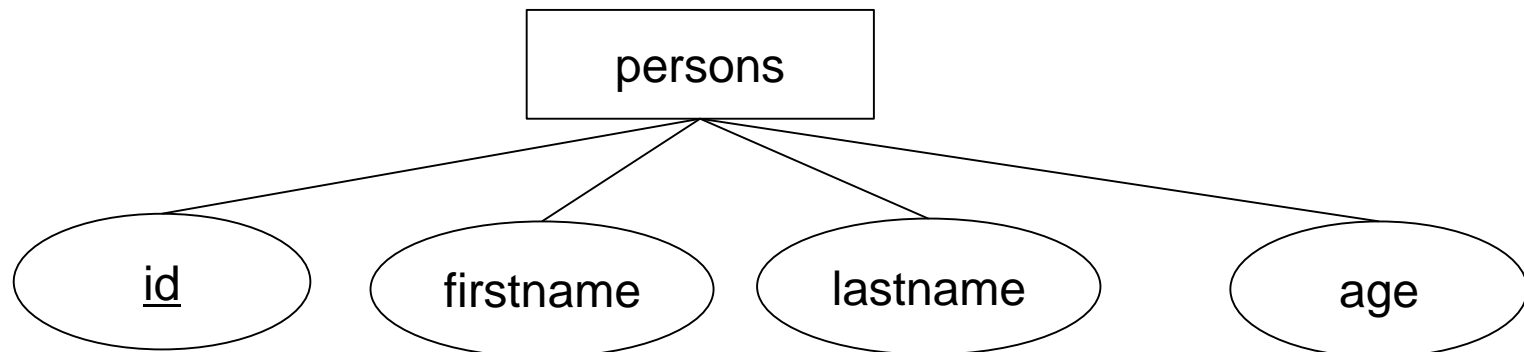
ALTER TABLE <table name> <specification>

Specification:

- ADD <col name> <col definition>
 - CHANGE <old col name> <new col name> <col definition>
 - MODIFY <col name> <col definition>
 - DROP <col name>
-
- Example:
ALTER TABLE courses ADD moodle boolean not null;

SQL Exercise 2

- Change the table named persons and add the column “firstname” and “lastname” using the ALTER TABLE statement
- Remove the column “name”
- Add a NOT NULL constraint to “age”



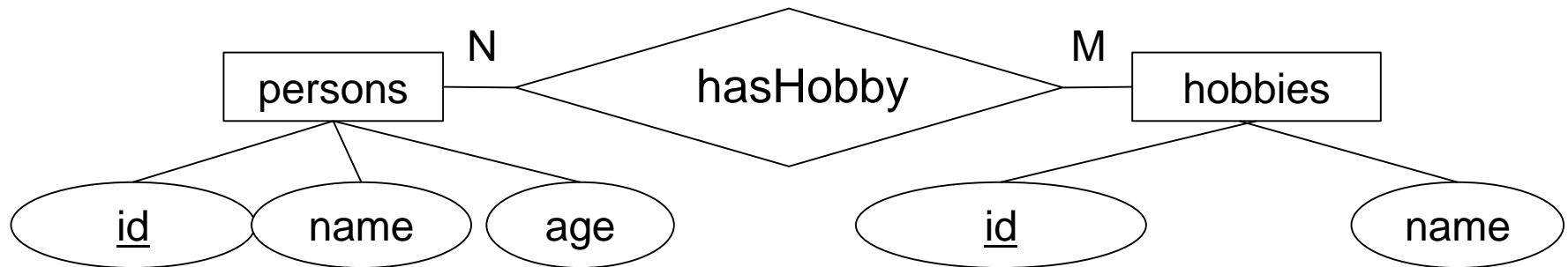
Change schema with SQL – RENAME TABLE

- Renames a table
- The syntax is as follows
 - `RENAME TABLE <table_name> TO <new_table_name>`
- SQL Exercise 3 rename your persons table to people

Delete schema with SQL - DROP TABLE

- Removes one or more tables
- The syntax is as follows
 - DROP TABLE <table_name>,...
- SQL Exercise 4 drop your people table!

SQL Exercise 5



- Create table **persons** with **name**, **age** and **id**
- Create a table **hobbies** with **id** and **name**
- Create a table **hasHobby** with **personId** and **hobbyId**
- Use appropriate data types
- Use the not null and primary key constraints

Data insertion with SQL – INSERT INTO

- The INSERT INTO statement is used to insert new records in a table
- Two ways to use:

```
INSERT INTO <table_name> VALUES (<value1>,<value2>,<value3>,...);
```

- Values in order of columns

```
INSERT INTO <table_name> (<column1>,<column2>,<column3>,...)  
VALUES (<value1>,<value2>,<value3>,...);
```

- Values in desired order

- Example:

```
INSERT INTO Courses VALUES(1,'EngDB','N0199');
```

Data retrieval with SQL - SELECT

- The SELECT statement retrieves data from tables
- The syntax is as follows:
SELECT <columns> FROM <data source> WHERE <condition> [...]
- Examples:

```
SELECT name FROM Courses
```

```
SELECT * FROM Courses
```

```
SELECT * FROM Courses WHERE Room='N0199'
```

```
SELECT * FROM Courses WHERE Room='M0199'
```

SQL Exercise 5

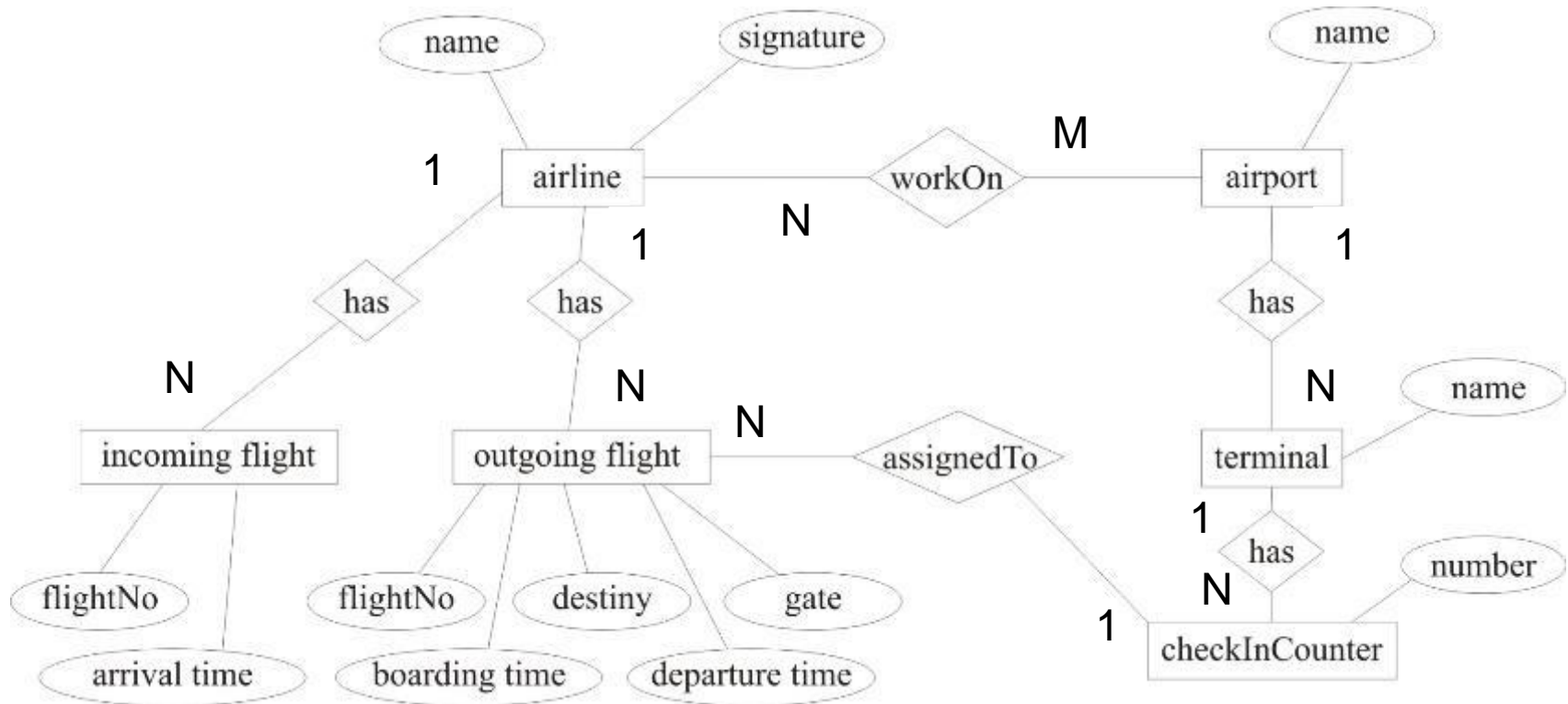
- Enter two students sitting near you into the persons table using the INSERT INTO statement
 - Once without column specification (the first one)
 - Once with columns in the order "name", "age", "id"
- Enter two hobbies

SQL Exercise 6

- Retrieve all the data you entered using the SELECT statement
- Retrieve a list of only the names
- Retrieve the age of the first student you entered using WHERE on the Name column

SQL Exercise 7 (Homework)

- Create the tables (including attributes and constraints) of the airline example



- Think about it: N:M (new table) but how to handle (1:N)?



End of Lecture

Thank you for your attention