Advanced SQL

Databases

Ira Assent

ira@cs.au.dk

Data-Intensive Systems group, Department of Computer Science, Aarhus University, DK

Intended learning outcomes

- Be able to
  - Write simple SQL DML queries
  - Make use of advanced constructs in DML queries

ira@cs.au.dk

Recap: SQL DDL

- DDL: define database schema in SQL
- `CREATE DATABASE Uni;`
- `CREATE TABLE Student (id INT PRIMARY KEY, name VARCHAR(15), office VARCHAR(10) REFERENCES Rooms(rooms));`
- Data types for numeric, string, date etc
- If you want quality (and you do), define constraints!
  - Keys: uniquely identify tuples
    - `Primary key`: within a table
    - `Foreign key`: across tables
- Other properties: `NOT NULL, UNIQUE, DEFAULT, triggered actions CASCADE, SET NULL, SET DEFAULT`
- Make use of documentation for definitions and examples

  https://dev.mysql.com/doc/refman/8.0/en/create-table.html

ira@cs.au.dk

Which SQL DDL for Group?

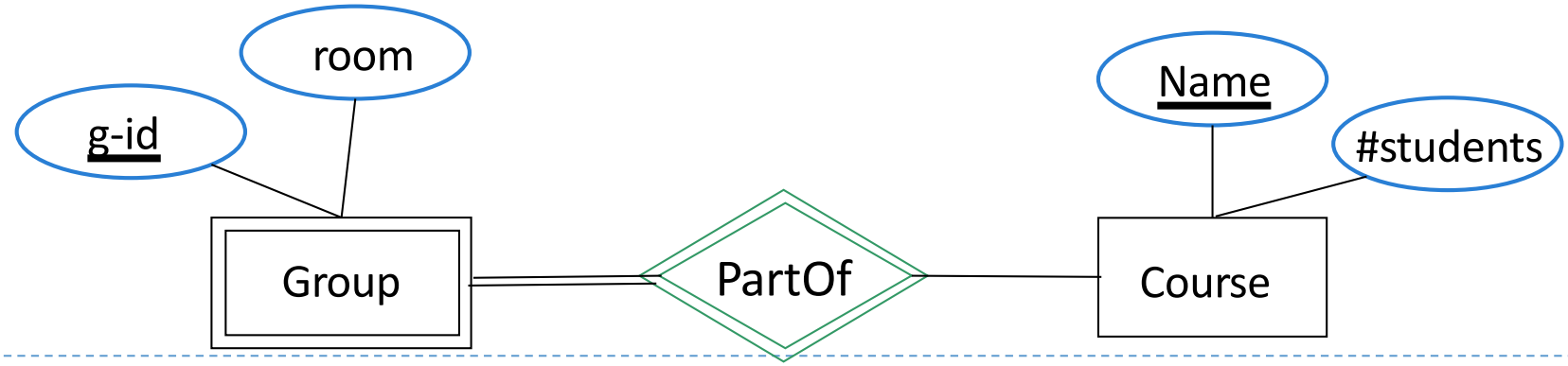
A. `CREATE TABLE GROUP(gid INT PRIMARY KEY, room VARCHAR(10));`

B. `CREATE TABLE GROUP(gid INT PRIMARY KEY, Name VARCHAR(15) PRIMARY KEY, room VARCHAR(10));`

C. `CREATE TABLE GROUP(gid INT, Name VARCHAR(15), room VARCHAR(10), PRIMARY KEY(gid,Name));`

D. `CREATE TABLE GROUP(gid INT PRIMARY KEY, Name VARCHAR(15), room VARCHAR(10), FOREIGN KEY(Name) REFERENCES Course(Name));`

E. `CREATE TABLE GROUP(gid INT, Name VARCHAR(15), room VARCHAR(10), PRIMARY KEY (gid, Name), FOREIGN KEY(Name) REFERENCES Course(Name));`



ira@cs.au.dk

SQL DML: SELECT-FROM-WHERE

- The basic form of an SQL query

```
SELECT  desired attributes
  FROM  one or more tables
  WHERE  condition about the involved rows
```

topic
DB
DB

Meetings

meetid	date	owner	topic
34716	2023-08-28	ira	DB
34717	2024-01-22	ira	DB
42835	2023-08-18	aas	Prog

- Which meetings ("topic") has ira arranged?

```
SELECT topic
  FROM Meetings
  WHERE owner = 'ira';
```

Loop Semantics for Single Table

- *Loop through all rows in the table*
- *Check if the condition is true*
- *Project the rows onto the desired attributes*

Note that duplicates are kept!

https://dev.mysql.com/doc/refman/8.0/en/select.html

ira@cs.au.dk

(Funny) terminology

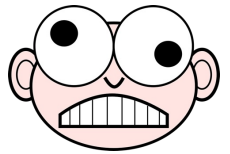
- The basic form of an SQL query

```
SELECT  desired attributes
  FROM  one or more tables
  WHERE  condition about the involved rows
```

Projection: projecting to attributes

Join: combining tables

Selection condition: selecting rows



- Which meetings ("topic") has ira arranged?

```
SELECT topic
  FROM Meetings
  WHERE owner = 'ira';
```

topic
DB
DB

Meetings

meetid	date	owner	topic
34716	2023-08-28	ira	DB
34717	2024-01-22	ira	DB
42835	2023-08-18	aas	Prog

ira@cs.au.dk

Combining conditions in WHERE

- AND, OR, NOT, =, <>, <, >, <=, >=, ...

```
SELECT owner, topic
  FROM Meetings
  WHERE owner ='ira' OR topic ='DB' ;
```

Meetings

meetid	date	owner	topic
34716	2023-08-28	ira	DB
34717	2024-01-22	ira	DB
42835	2023-08-18	aas	Prog

owner	topic
ira	DB
Ira	DB

ira@cs.au.dk

Renaming in SELECT

- The selected attributes can be given new names

```
SELECT name AS navn, office AS kontor
  FROM People
  WHERE office = 'Ny-357';
```

userid	name	office
ira	Ira Assent	Ny-357
aas	Annika Schmidt	NULL
jan	Jan Christensen	Ho-017

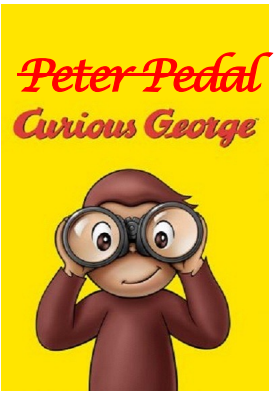
navn	kontor
Ira Assent	vip

ira@cs.au.dk

Renaming in FROM

- Example

```
SELECT name,office
      FROM People AS Folk
  WHERE Folk.office = 'Ny-357';
```



userid	name	office
ira	Ira Assent	Ny-357
aas	Annika Schmidt	NULL
jan	Jan Christensen	Ho-017

name	office
Ira Assent	vip

ira@cs.au.dk

General renaming options

- Specifying a new name right after the original one using AS called an **alias** or **tuple variable**
  - Declare alternative relation name `E` instead of `Employee`
  - `SELECT Fname FROM EMPLOYEE AS E;`
    - Has the same attributes and entries
  - `SELECT Fn FROM EMPLOYEE AS E(Fn, Mi, Ln, Ssn, Bd, Addr, Sex, Sal, Sssn, Dno);`
    - Also changes the attribute names

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

ira@cs.au.dk

Math in SELECT

- The attributes may have computed values

```
SELECT customer AS kunde, date AS dato, price*7.44 AS pris
  FROM Purchase
  WHERE customer = 'ira';
```

Allows us to convert the price from EUR to DKK

id	date	price	customer	product
16	2024-02-03	2.99	ira	Sencha
17	2024-01-22	1.59	ira	Müsli
5	2023-12-18	6.49	aas	Arabica

kunde	dato	pris
ira	2024-02-03	22.2456
ira	2024-01-22	11.8296

ira@cs.au.dk

Arithmetic Operators

- Standard arithmetic operators:
  - Addition (+), subtraction (–), multiplication (*), and division (/)
  - Convenient both in SELECT clause and in WHERE clause

```
SELECT customer AS PricelessCustomer
  FROM Purchase
  WHERE fee + price > 7;
```

id	fee	price	customer
16	0.15	2.99	ira
17	2.5	1.59	ira
5	1.75	6.49	aas

customer
aas

- `BETWEEN` comparison operator
  - Returns values that are greater than or equal to first number and smaller than or equal to second number

```
SELECT price AS Sweetspot
  FROM Purchase
  WHERE price BETWEEN 2.99 AND 4.99;
```

price
2.99

ira@cs.au.dk

Multiple Relations / Joins

- Who has booked meetings on January 22, 2024?

```
SELECT name
  FROM People, Meetings
  WHERE date = '2024-01-22'
  AND owner = userid;
```

- Relations are **joined**
  - **Join condition** `owner=userid` states how attributes from joined tables should match

General Loop Semantics

- *Loop through all rows in all tables*
- *For each combination*
  - *check if the condition is true*
  - *project the rows onto the desired attributes*
- *Note that duplicates are still kept*

People

userid	name	group	office
ira	Ira Assent	vip	Ny-357
aas	Annika Schmidt	phd	NULL
jan	Jan Christensen	tap	Ho-017

Meetings

meetid	date	owner	topic
34716	2023-08-28	ira	dDB
34717	2024-01-22	ira	dDB
42835	2023-08-18	aas	TA meeting

name
Ira Assent

ira@cs.au.dk

Example Company database from textbook

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

ira@cs.au.dk

Simple queries on the example

Query 0. Retrieve the birth date and address of the employee(s) whose name is 'John B. Smith'.

```
Q0:   SELECT   Bdate, Address
      FROM     EMPLOYEE
      WHERE    Fname='John' AND Minit='B' AND Lname='Smith';
```

Bdate	Address
1965-01-09	731Fondren, Houston, TX

Query 1. Retrieve the name and address of all employees who work for the 'Research' department.

```
Q1:   SELECT   Fname, Lname, Address
      FROM     EMPLOYEE, DEPARTMENT
      WHERE    Dname='Research' AND Dnumber=Dno;
```

Fname	Lname	Address
John	Smith	731 Fondren, Houston, TX
Franklin	Wong	638 Voss, Houston, TX
Ramesh	Narayan	975 Fire Oak, Humble, TX
Joyce	English	5631 Rice, Houston, TX

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire

ira@cs.au.dk

Joining more tables

Query 2. For every project located in 'Stafford', list the project number, the controlling department number, and the department manager's last name, address, and birth date.

```
Q2:   SELECT   Pnumber, Dnum, Lname, Address, Bdate
      FROM     PROJECT, DEPARTMENT, EMPLOYEE
      WHERE    Dnum=Dnumber AND Mgr_ssn=Ssn AND
               Plocation='Stafford';
```

Pnumber	Dnum	Lname	Address	Bdate
10	4	Wallace	291Berry, Bellaire, TX	1941-06-20
30	4	Wallace	291Berry, Bellaire, TX	1941-06-20

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire

ira@cs.au.dk

Which of these is correct?

1. `SELECT name FROM Meetings, Rooms WHERE capacity = '6' AND room = office;`
2. `SELECT Meetings, Rooms WHERE capacity = '6' AND room = office;`
3. `SELECT name FROM Meetings AND Rooms WHERE capacity = '6' AND room = office;`
4. `SELECT name FROM Meetings WHERE capacity = '6' AND room = office;`

userid	name	group	office
ira	Ira Assent	vip	Ny-357
aas	Annika Schmidt	phd	NULL
jan	Jan Christensen	tap	Ho-017

Meetings

room	capacity
Ny-357	6
Ada-333	26
StoreAud	286

Rooms

ira@cs.au.dk

Ambiguous Attribute Names

- Same name can be used for two (or more) attributes
  - As long as the attributes are in different relations
    - In practice very common: e.g. attribute names id, name in Student (id, name, studies, city), Professor (id, name, department, teaches)
  - If using two same name attributes in the same query, must **qualify** attribute name with relation name to prevent ambiguity
    - `Relation.Attribute`
    - Else get error message

```
SELECT   Fname, EMPLOYEE.Name, Address
FROM     EMPLOYEE, DEPARTMENT
WHERE    DEPARTMENT.Name='Research' AND
         DEPARTMENT.Dnumber=EMPLOYEE.Dnumber;
```


ira@cs.au.dk

Unspecified WHERE Clause

- Missing `WHERE` clause
  - Indicates no condition on tuple selection
    - Thus, all tuples in relation returned
    - Use as quick view of your table contents!

  - When querying two or more tables returns the cross product
    - All possible combinations of tuples from either table
      - i.e., a lot!!

```
SELECT    Ssn
FROM      EMPLOYEE;

SELECT    Ssn, Dname
FROM      EMPLOYEE, DEPARTMENT;
```


ira@cs.au.dk

```
SELECT  Ssn
FROM    EMPLOYEE;

SELECT  Ssn, Dname
FROM    EMPLOYEE, DEPARTMENT;
```

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	
John	B	Smith	123456789	1965-01-09	7
Franklin	T	Wong	333445555	1955-12-08	6
Alicia	J	Zelaya	999887777	1968-01-19	3
Jennifer	S	Wallace	987654321	1941-06-20	2
Ramesh	K	Narayan	666884444	1962-09-15	9
Joyce	A	English	453453453	1972-07-31	5
Ahmad	V	Jabbar	987987987	1969-03-29	9
James	E	Borg	888665555	1937-11-10	4

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mg
Research	5	333445555	19
Administration	4	987654321	19
Headquarters	1	888665555	19

Ssn	Dname
123456789	Research
333445555	Research
999887777	Research
987654321	Research
666884444	Research
453453453	Research
987987987	Research
888665555	Research
123456789	Administration
333445555	Administration
999887777	Administration
987654321	Administration
666884444	Administration
453453453	Administration
987987987	Administration
888665555	Administration
123456789	Headquarters
333445555	Headquarters
999887777	Headquarters
987654321	Headquarters
666884444	Headquarters
453453453	Headquarters
987987987	Headquarters
888665555	Headquarters

SSN
123456789
333445555
999887777
987654321
666884444
453453453
987987987
888665555

PT_LOCATIONS

number	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

IENT;

How many rows if you do cross product on three tables with 10 tuples each?

ira@cs.au.dk

Asterisk notation

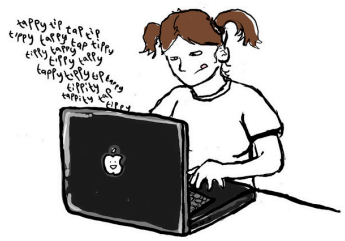
- Specify an asterisk * instead of attributes in SELECT clause
  - Retrieve all the attribute values of the selected tuples
  - Is the same as listing all attributes of the relation(s)

```
SELECT * FROM EMPLOYEE;
```

same as

```
SELECT (Fname, Minit, Lname, Ssn, Bdate, Address,
Sex, Salary, Super_ssn, Dno) FROM EMPLOYEE;
```

  - Saves a lot of typing



EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4

Using the same relation twice (self-join)

Find all pairs of roommates like (Annika, Ira)

Do not produce pairs with self like (Ira, Ira)

Do not produce duplicates like (Ira, Annika) for (Annika, Ira) – use alphabetic order

1. ```
   SELECT name, name FROM People WHERE office =
   office  AND name = name;
   ```
2. ```
   SELECT name, name FROM People WHERE (name,
   office);
   ```
3. ```
   SELECT p1.name, p2.name FROM People AS p1,
   People AS p2 WHERE p1.office = p2.office
      AND p1.name < p2.name;
   ```
4. ```
   SELECT p1.name, p2.name FROM People p1,
   People p2 WHERE p1.office = p2.office;
   ```


ira@cs.au.dk

Finding a pattern in a string

We would like to find all events relating to beer that are organized by Frank

```
SELECT owner, topic
  FROM Meetings
  WHERE owner ='fra' AND topic LIKE '%beer%';
```

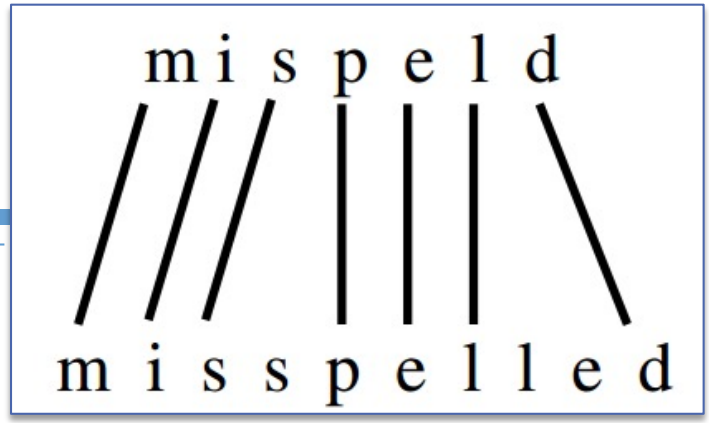
owner	topic
fra	afternoon beer
fra	beer tasting

Meetings

meetid	date	owner	topic
34716	2023-08-28	ira	dDB
34717	2024-01-22	ira	dDB
42835	2023-10-18	aas	TA meeting
43779	2024-02-01	fra	afternoon beer
48333	2024-02-08	fra	beer tasting
50001	2024-01-09	joe	return beer bottles

ira@cs.au.dk

Fun with Pattern Matching



- **`LIKE`** comparison operator
  - Used for string **pattern matching**
  - `%` replaces an arbitrary number of zero or more characters
  - underscore _ replaces a single character

```
SELECT Name FROM Student WHERE Address LIKE '%Aarhus%';
```

  - Matches *Aarhus*, *8200 Aarhus N*, *8000 Aarhus*, etc.

```
SELECT Name FROM Student WHERE Address LIKE 'Aarhus__';
```

  - Matches *Aarhus N*, but NOT *Aarhus*!
- If you need to find a string with %, precede it with an escape character \

```
SELECT comment FROM Code WHERE comment LIKE '\ % \ % \ %%';
```

  - Matches *%%%My Comment%%%*

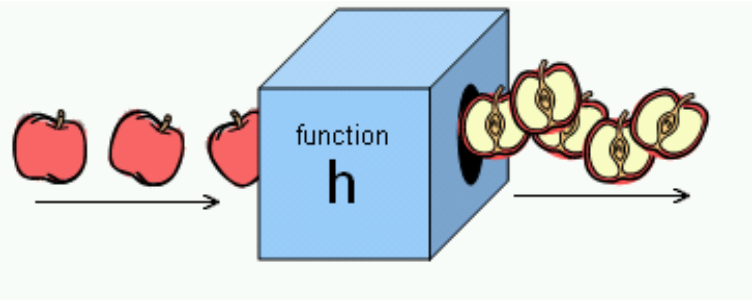
```
SELECT CONCAT(FirstName, ' ', LastName) AS FullName FROM
Employees;
```

Concatenates first name, a space, and the last name into a single return value

ira@cs.au.dk

Scalar functions

- May use scalar functions
- Many available
  - integer and float functions
  - string functions
  - calendar functions, ...




```
SELECT CHARACTER_LENGTH(name) as len,
UPPER(`group`) as `GROUP` FROM People;
```

userid	name	group	office
ira	Ira Assent	vip	Ny-357
aas	Annika Schmidt	phd	NULL
jan	Jan Christensen	tap	Ho-017

len	GROUP
9	VIP
14	PHD
15	TAP

https://dev.mysql.com/doc/refman/8.0/en/built-in-function-reference.html

ira@cs.au.dk

Tables as Sets in SQL

- SQL does not automatically eliminate duplicate tuples in query results
  - i.e., uses **multiset semantics**
- Use the keyword **`DISTINCT`** in the `SELECT` clause
  - Turns result into **set**
  - Each value remains only once in the result
  - Expensive operation
- Which studies do our students follow?

```
SELECT studies FROM Student;
SELECT DISTINCT studies FROM Student;
```

id	name	studies
1	Joe	CS
2	Jane	ITP
3	Alice	CS
4	Bob	CE
5	Eve	DS
6	Bo	CS
7	Ib	ITP
8	Liv	ITP

studies
CS
ITP
CS
CE
DS
CS
ITP
ITP

studies
CS
ITP
CE
DS

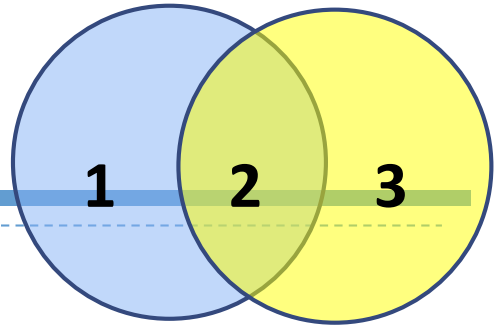
If not distinct, can specify **`ALL`**

```
SELECT  ALL Salary
FROM    EMPLOYEE;

SELECT  DISTINCT Salary
FROM    EMPLOYEE;
```


ira@cs.au.dk

Set Operations in SQL



- Set operations
  - `1 UNION 3` : elements in 1 or in 3,
  - `1 EXCEPT 3` : elements in1 but not in 3 (difference),
  - `1 INTERSECT 3` : elements in 1 and 3 : 2

mathematical set operations ∪, \ and ∩

```
SELECT name FROM Student
UNION
SELECT name FROM TA
```

~~`SELECT * FROM Student`~~
~~`UNION`~~
~~`SELECT * FROM TA`~~

Works only if the input relations have identical schema!

Removes duplicates (set, not multi-set)

id	name	studies
1	Joe	CS
2	Jane	ITP
3	Alice	CS
4	Bob	CE

name
Joe
Jane
Alice
Bob
Eve
Chris

TAid	name	course
1	Eve	DB
2	Chris	DB
2	Chris	Prog

ira@cs.au.dk

UNION, INTERSECT, and EXCEPT

- Corresponding multiset operations: `UNION ALL`, `EXCEPT ALL`, `INTERSECT ALL`
- Uses multiset semantics
  - i.e., keep duplicates

```
SELECT name FROM Student
UNION ALL
SELECT name FROM TA
```

id	name	studies
1	Joe	CS
2	Jane	ITP
3	Alice	CS
4	Bob	CE

TAid	name	course
1	Eve	DB
2	Chris	DB
2	Chris	Prog

name
Joe
Jane
Alice
Bob
Eve
Chris
Chris

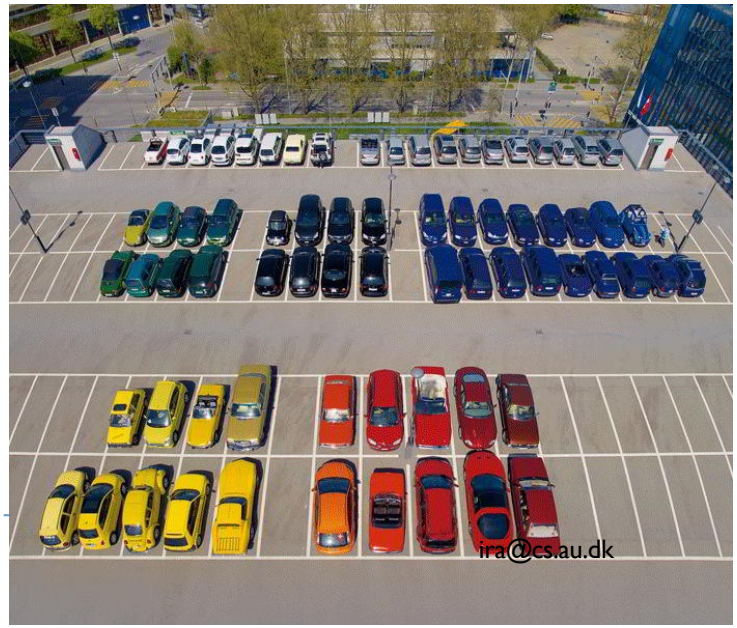
  - Different semantics:
    - Set: find names of all students or TAs
    - Multi-set: find names of students or TAs the number of times they occur

https://dev.mysql.com/doc/refman/8.0/en/union.html
https://dev.mysql.com/doc/refman/8.0/en/intersect.html
https://dev.mysql.com/doc/refman/8.0/en/except.html

ira@cs.au.dk

Ordering of Query Results

- Use **`ORDER BY`** clause
  - Keyword **`DESC`** to see result in a descending order of values
    - 10,9,8,7,…
    - Zebra, Mouse, Elephant,…
  - Keyword **`ASC`** to specify ascending order explicitly (default)
  - `ORDER BY D.Dname DESC, E.Lname ASC, E.Fname ASC`
    - 1,2,3,…
    - Ape, Bear, Chimp,…



Modifications

- SQL **commands / statements** may modify the database
  - Not "query"

- Three kinds of modifications
  - insert one or more rows
  - delete one or more rows
  - update existing rows or columns

- Modifications do not return a result
  - Their effect is on the database state
  - E.g. new price value for a product, new student added, old computer removed

ira@cs.au.dk

Inserting a Single Row

- INSERT INTO *table* VALUES (*list of values*);

```
INSERT INTO Participants
  VALUES (42835, 'ira', 'a');
```

- Optionally specify attribute names:

```
INSERT INTO
  Participants(pid, status, meetid)
  VALUES ('ira', 'a', 42835);
```

- Missing values are `NULL` or defaults

https://dev.mysql.com/doc/refman/8.0/en/insert.html

ira@cs.au.dk

Example INSERTs from textbook

- Specify the relation name and a list of values for the tuple (assumes attributes and their order is known and obeyed)

```
INSERT INTO    EMPLOYEE
VALUES         ( 'Richard', 'K', 'Marini', '653298653', '1962-12-30', '98
                 Oak Forest, Katy, TX', 'M', 37000, '653298653', 4 );
```

- To insert only a subset of the values, or to specify the order (assumes remaining attributes admit NULL values or have default values):

```
INSERT INTO EMPLOYEE(Fname, Lname, Dno, Ssn)
VALUES ('Richard', 'Marini', 4, '653298653');
```


ira@cs.au.dk

The UPDATE Command

- Modify attribute values of one or more selected tuples
- Additional **`SET`** clause in the **`UPDATE`** command
  - Specifies attributes to be modified and new values

```
UPDATE    PROJECT
SET       Plocation = 'Bellaire', Dnum = 5
WHERE     Pnumber=10;
```

https://dev.mysql.com/doc/refman/8.0/en/update.html

ira@cs.au.dk

Update example

- UPDATE *table* SET *attribute assignments*
  WHERE *condition*;

- Move Ira to a smaller office

userid	name	group	office
ira	Ira Assent	vip	Ny-357
aas	Annika Schmidt	phd	NULL
jan	Jan Christensen	tap	Ho-017

```
UPDATE People
  SET office = 'Ny-343'
  WHERE userid = 'ira';
```

userid	name	group	office
ira	Ira Assent	vip	Ny-343
aas	Annika Schmidt	phd	NULL
jan	Jan Christensen	tap	Ho-017

ira@cs.au.dk

The DELETE Command

- Removes tuples from a relation
  - Includes a `WHERE` clause to select the tuples to be deleted

```
DELETE FROM     EMPLOYEE
WHERE           Lname='Brown';

DELETE FROM     EMPLOYEE
WHERE           Ssn='123456789';

DELETE FROM     EMPLOYEE
WHERE           Dno=5;

DELETE FROM     EMPLOYEE;
```

Careful!
Practice tip: try out the condition as part of a select first to see if you get the right tuples, then issue the delete with the same condition

https://dev.mysql.com/doc/refman/8.0/en/delete.html

ira@cs.au.dk

Deleting Some Rows

room	capacity
Ny-357	6
Ada-333	26
StoreAud	286

- DELETE FROM *table* WHERE *condition*;
- Delete Ira's office

```
DELETE FROM Rooms
  WHERE room='Ny-357';
```

room	capacity
Ada-333	26
StoreAud	286

- Delete all offices

```
DELETE FROM Rooms;
```

room	capacity

ira@cs.au.dk

Summary

- Intended learning outcomes
  - Be able to
    - Write simple SQL DML queries
    - Make use of advanced constructs in DML queries

ira@cs.au.dk

Where to go from here?

- So, we can use SQL to create a database schema, to insert and manipulate data
  - But, how do we ask more complicated questions?
    - How can I check if something is found in the database?
      - E.g. is there are student who has finished all courses or a professor who does not teach anything?
    - What if I want to make use of query results in another query?
    - We can express very powerful conditions in SQL using subqueries
      - SQL full power coming up!

ira@cs.au.dk

What was this all about?

Guidelines for your own review of today's session

- A standard SQL DML query is…
  - We can specify constraints…
  - It is evaluated in the following manner…
  - A join is when…
  - We specify conditions using…
  - We order results…
- We use aggregates, do pattern matching and arithmetics…
- We can choose to remove or keep duplicates and avoid name clashes by…
- SQL handles data as multisets which are…
  - Set operations are…
- We can insert, delete and update as follows…

ira@cs.au.dk