AALBORG
UNIVERSITY

Database System (SW5)

2. Relational Model

Tiantian Liu
Department of Computer Science
Aalborg University
Fall 2025

Motivation

AALBORG UNIVERSITY

- SQL's data model resembles the relational model
- SQL is partly based on relational algebra
- Relational algebra helps understanding query plans and query optimization

Learning Goals

AALBORG UNIVERSITY

- Explain the relational model
- Create non-trivial relational algebra queries
- Use the various join types in relational algebra queries

Agenda

- Relational Model
  - Relation, tuple, attribute
  - Schema and instance
  - Keys: Superkey, candidate key, primary key, foreign key
- Relational Algebra

-Tiantian Liu

Relation

AALBORG UNIVERSITY

- In the relational model the term **relation** is used to refer to a table

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

- The *Instructor* Relation

Tuple

AALBORG UNIVERSITY

- The term **tuple** is used to refer to a row.

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

- The *Instructor* Relation

Relations are Unordered

AALBORG UNIVERSITY

- Order of tuples is irrelevant (tuples may be stored in an arbitrary order)

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

Attribute

AALBORG UNIVERSITY

- The term **attribute** refers to a column of a table

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

- The *Instructor* Relation
- Attribute: *ID, name, dept_name, salary*

Attribute--Domain

AALBORG UNIVERSITY

- For each attribute of a relation, there is a set of permitted values, called the **domain** of that attribute.

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

- The *Instructor* Relation
- The domain of the *dept_name* attribute is the set of all possible department names.

Atomic Domain

AALBORG UNIVERSITY

- Why should a domain be atomic?

ID	Name	Address
001	Elodie	59, SQL Street
002	Felix	103, HTML Road
003	Soren	32, PHP Street

- Domain of *Address* attribute
  - (59, SQL Street), (103, HTML Road), (32, PHP Street)

Atomic Domain

AALBORG UNIVERSITY

- Better Table

ID	Name	Street_num	Street_name
001	Elodie	59	SQL Street
002	Felix	103	HTML Road
003	Soren	32	PHP Street

- When the data is atomic, that means the data has been broken down into the smallest pieces that can't or shouldn't be divided.
  - Not necessarily smallest possible.
- Again, this depends on your requirement on getting information.
  - Being atomic in one scenario may not be so in another.

Relation Schema and Instance

AALBORG UNIVERSITY

- $A_1, A_2, \ldots, A_n$ are attributes
- $R = (A_1, A_2, \ldots, A_n)$ is a relation schema

  Example

  instructor = (*ID, name, dept_name, salary*)
- A relation instance *r* defined over schema *R* is denoted by *r*(*R*).
- An element *t* of relation *r* is called a tuple and is represented by a row in a table.

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

Relation instance

Database Schema

AALBORG UNIVERSITY

- Database schema -- is the logical structure of the database.
- Database instance -- is a snapshot of the data in the database at a given instant in time.
- Example: University Database
  - *instructor* (*ID, name, dept_name, salary*)
  - *course* (*course_id, title, dept_name ,credits*)
  - *department* (*dept_name, building, budget*)
  - *section* (*course_id, sec_id, semester, year, building, room_number, time_slot_id*)
  - *teaches* (*ID, course_id, sec_id, semester, year*)
  - *student* (*ID, name, dept_name, tot_cred*)
  - *advisor* (*s_id, i_id*)
  - *takes* (*ID, course_id, sec_id, semester, year, grade*)
  - *classroom* (*building, room_number, capacity*)
  - *time slot* (*time_slot_id, day, start_time, end_time*)

Superkey

AALBORG UNIVERSITY

- Let $K \subseteq R$
- K is a superkey of R if values for K are sufficient to identify a unique tuple of each possible relation $r(R)$
- If t_1 and t_2 are in r and $t_1 \neq t_2$, then $t_1.K \neq t_2.K$.
- Example: *instructor* (*ID, name, dept_name, salary*)
  - {*ID*}
  - {*name*}
  - {*dept_name*}
  - {*salary*}
  - {*ID, name*}
  - {*ID, name, dept_name*}
  - {*dept_name, salary*}
  - {*ID, name, dept_name, salary*}

Candidate key

AALBORG UNIVERSITY

- Superkey *K* is a **candidate key** if *K* is minimal
  - A superkey may contain extraneous attributes.
  - We are often interested in superkeys for which no proper subset is a superkey.
- Example: *instructor* (*ID, name, dept_name, salary*)
  - {*ID*}, {*ID, name*}, {*ID, name, dept_name*} are superkey
  - {*ID*} is a candidate key for *Instructor*
  - {*ID, name*}, {*ID, name, dept_name*} *are not candidate keys*
- Suppose that a combination of *name* and *dept_name* is sufficient to distinguish among members of the instructor relation
  - Both {*ID*} and {*name, dept_name*} are candidate keys

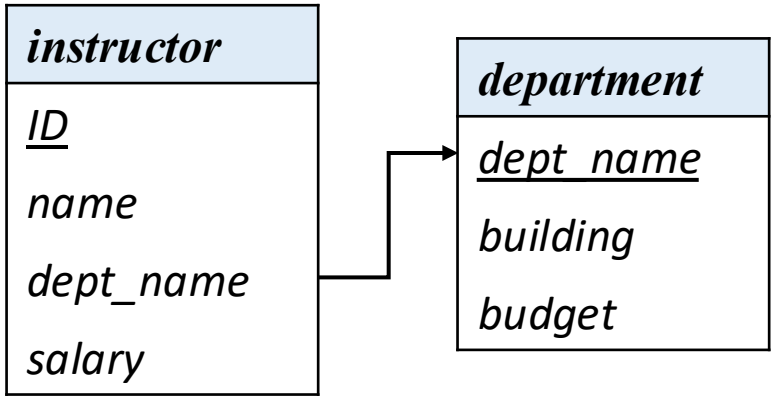
Primary key

AALBORG UNIVERSITY

- One of the candidate keys is selected to be the primary key.
- Which one?
  - Primary keys must be chosen with care
  - The primary key should be chosen such that its attribute values are never, or are very rarely, changed
- Examples
  - *instructor* (*ID*, *name, dept_name, salary*)
  - *course* (*course_id*, *title, dept_name ,credits*)
  - *department* (*dept_name*, *building, budget*)
  - *student* (*ID*, *name, dept_name, tot_cred*)
  - *classroom* (*building, room_number*, *capacity*)

Foreign key

- Foreign key constraint: Value in one relation must appear in another
- Example:
  - *instructor* (*ID*, *name*, *dept_name*, *salary*)
  - *department* (*dept_name*, *building*, *budget*)
  - *dept_name* in *instructor* is a foreign key from *instructor* referencing *department*
  - Referencing relation: *instructor*
  - Referenced relation: *department*



Summary--Keys

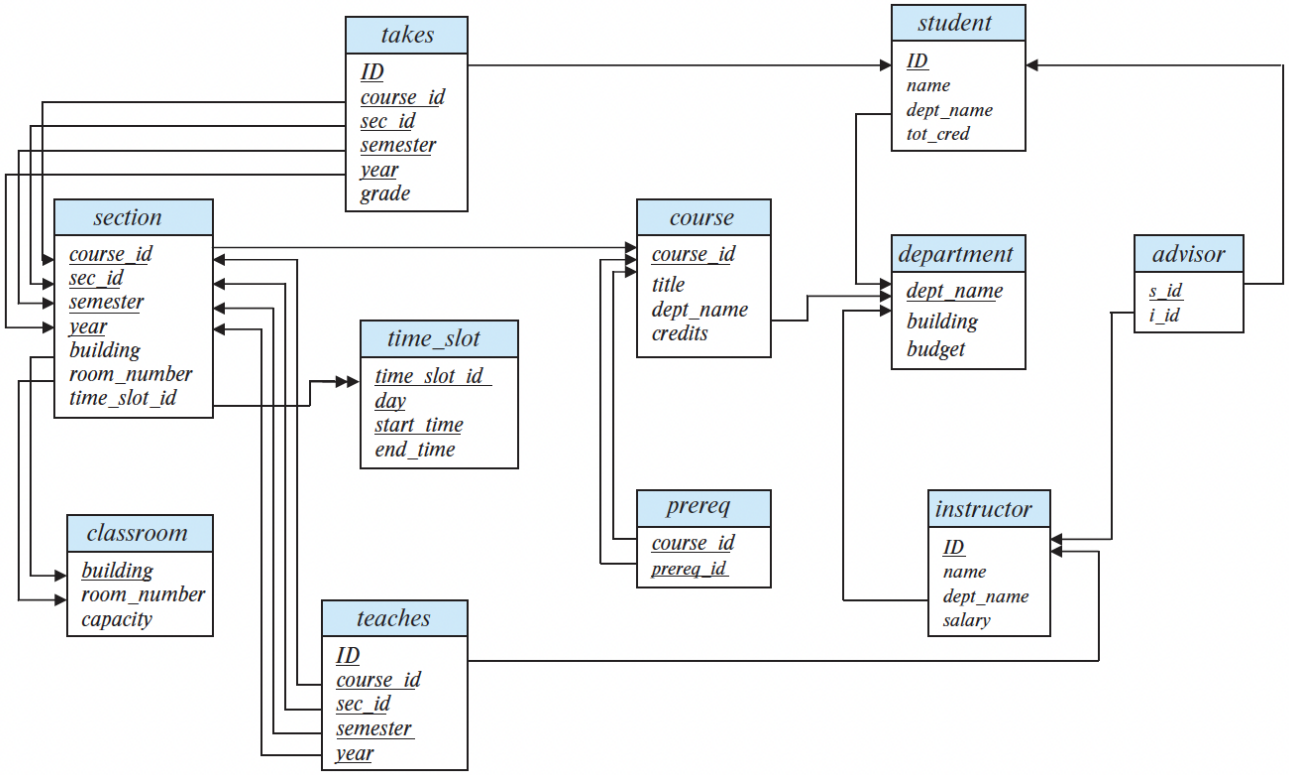
AALBORG UNIVERSITY

- Superkey
- Candidate key
- Primary key
- Foreign key

Schema Diagram

AALBORG UNIVERSITY

- A database schema, along with primary key and foreign-key constraints, can be depicted by schema diagrams



Agenda

- Relational Model
  - Relation, tuple, attribute
  - Schema and instance
  - Keys: Superkey, candidate key, primary key, foreign key
- Relational Algebra

-Tiantian Liu

Relational Algebra

AALBORG UNIVERSITY

- Relational Algebra: A language consisting of a set of operations that take one or two relations as input and produce a new relation as their result.
- Six basic operators
  - select: σ
  - project: Π
  - union: $\cup$
  - set difference: –
  - Cartesian product: x
  - rename: ρ

Select Operation

AALBORG UNIVERSITY

- The **select** operation selects tuples that satisfy a given predicate.
- Notation: σ_p (r)
  - *p* is called the **selection predicate**
- Example: select those tuples of the *instructor* relation where the instructor is in the "Physics" department.
- Query

$$\sigma_{dept\_name = \text{“Physics”}} (instructor)$$

Select Operation

AALBORG UNIVERSITY

- Example: select those tuples of the *instructor* relation where the instructor is in the "Physics" department.
- $\sigma_{dept\_name="Physics"}(instructor)$

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

Result

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
33456	Gold	Physics	87000

Select Operation

AALBORG UNIVERSITY

- Example: select the instructor named “Kim”.
- $\sigma_{name=\text{“Kim”}}(instructor)$

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

Result

ID	*name*	*dept_name*	*salary*
98345	Kim	Elec. Eng.	80000

Select Operation

AALBORG UNIVERSITY

- We allow comparisons using $=, \neq, >, \geq, <, \leq$ in the selection predicate.
- Example: select the instructors whose salary is greater than 80000
- $\sigma_{salary > 80000}(instructor)$

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

Result

ID	*name*	*dept_name*	*salary*
12121	Wu	Finance	90000
22222	Einstein	Physics	95000
33456	Gold	Physics	87000
83821	Brandt	Comp. Sci.	92000

Select Operation

AALBORG UNIVERSITY

- We can combine several predicates into a larger predicate by using the connectives: $\wedge$ **(and)**, $\vee$ **(or)**, $\neg$ **(not)**
- Example: select the Physics instructors whose salary is greater than 90000
- $\sigma_{dept\_name = \text{“Physics”} \wedge salary > 90000}(instructor)$

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

Result

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000

Project Operation

AALBORG UNIVERSITY

- A unary operation that returns its argument relation, with certain attributes left out.
- Notation:

$$\Pi_{A1,A2,A3\ldots.Ak}(r)$$

where A_1, A_2, …, A_k are attribute names and r is a relation name.

- The result is defined as the relation of k columns obtained by erasing the columns that are not listed
- Duplicate rows removed from result, since relations are sets

Project Operation

AALBORG UNIVERSITY

- Example: return the *dept_name* attribute of the *instructor*.
- $\Pi_{dept\_name}(instructor)$

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

Result

dept_name
Comp. Sci.
Finance
Music
Physics
History
Biology
Elec. Eng.

Composition of Relational Operations

AALBORG UNIVERSITY

- Consider the query -- Find the names of all instructors in the Physics department.
- $\Pi_{name}(\sigma_{dept\_name = \text{“Physics”}}(instructor))$

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

Result

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
33456	Gold	Physics	87000

name
Einstein
Gold

Composition of Relational Operations

AALBORG UNIVERSITY

- Query -- Find the names of all instructors in the Physics department.
- Think about this: $\sigma_{dept\_name = \text{“Physics”}} (\prod_{name}(instructor))$

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

Wrong

- The projection goes before the selection here
- Since the projection eliminates “*dept_name*’, the selection cannot be performed

Exercise 1

AALBORG UNIVERSITY

- Write the following query in relational algebra and think about the result.
- Query -- Find the IDs of all instructors whose salary is less than 82000.

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

Exercise 1--Solution

AALBORG UNIVERSITY

- Query -- Find the IDs of all instructors whose salary is less than 82000.

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

$$\Pi_{ID}(\sigma_{salary < 82000}(instructor))$$

Cartesian-Product Operation

AALBORG UNIVERSITY

- The Cartesian-product operation (denoted by X) allows us to combine information from any two relations.
- Example: the Cartesian product of the relations *instructor* and t*eaches* is written as:

$$instructor \times teaches$$

- *instructor* (*ID, name, dept_name, salary*)
- *teaches* (*ID, course_id, sec_id, semester, year*)

Cartesian-Product Operation

AALBORG UNIVERSITY

instructor X *teaches*

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

instructor

ID	*course_id*	*sec_id*	*semester*	*year*
10101	CS-101	1	Fall	2017
10101	CS-315	1	Spring	2018
10101	CS-347	1	Fall	2017
12121	FIN-201	1	Spring	2018
15151	MU-199	1	Spring	2018
22222	PHY-101	1	Fall	2017
32343	HIS-351	1	Spring	2018
45565	CS-101	1	Spring	2018
45565	CS-319	1	Spring	2018
76766	BIO-101	1	Summer	2017
76766	BIO-301	1	Summer	2018
83821	CS-190	1	Spring	2017
83821	CS-190	2	Spring	2017
83821	CS-319	2	Spring	2018
98345	EE-181	1	Spring	2017

teaches

Cartesian-Product Operation

AALBORG UNIVERSITY

instructor.ID	*name*	*dept_name*	*salary*	*teaches.ID*	*course_id*	*sec_id*	*semester*	*year*
10101	Srinivasan	Comp. Sci.	65000	10101	CS-101	1	Fall	2017
10101	Srinivasan	Comp. Sci.	65000	10101	CS-315	1	Spring	2018
10101	Srinivasan	Comp. Sci.	65000	10101	CS-347	1	Fall	2017
10101	Srinivasan	Comp. Sci.	65000	12121	FIN-201	1	Spring	2018
10101	Srinivasan	Comp. Sci.	65000	15151	MU-199	1	Spring	2018
10101	Srinivasan	Comp. Sci.	65000	22222	PHY-101	1	Fall	2017
...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...
12121	Wu	Finance	90000	10101	CS-101	1	Fall	2017
12121	Wu	Finance	90000	10101	CS-315	1	Spring	2018
12121	Wu	Finance	90000	10101	CS-347	1	Fall	2017
12121	Wu	Finance	90000	12121	FIN-201	1	Spring	2018
12121	Wu	Finance	90000	15151	MU-199	1	Spring	2018
12121	Wu	Finance	90000	22222	PHY-101	1	Fall	2017
...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...
15151	Mozart	Music	40000	10101	CS-101	1	Fall	2017
15151	Mozart	Music	40000	10101	CS-315	1	Spring	2018
15151	Mozart	Music	40000	10101	CS-347	1	Fall	2017
15151	Mozart	Music	40000	12121	FIN-201	1	Spring	2018
15151	Mozart	Music	40000	15151	MU-199	1	Spring	2018
15151	Mozart	Music	40000	22222	PHY-101	1	Fall	2017
...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...
22222	Einstein	Physics	95000	10101	CS-101	1	Fall	2017
22222	Einstein	Physics	95000	10101	CS-315	1	Spring	2018
22222	Einstein	Physics	95000	10101	CS-347	1	Fall	2017
22222	Einstein	Physics	95000	12121	FIN-201	1	Spring	2018
22222	Einstein	Physics	95000	15151	MU-199	1	Spring	2018
22222	Einstein	Physics	95000	22222	PHY-101	1	Fall	2017
...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...

Theta Join Operation

AALBORG UNIVERSITY

- The Cartesian-Product

  instructor X teaches

  associates every tuple of instructor with every tuple of teaches.
  - Most of the resulting rows have information about instructors who did NOT teach a particular course.
- To get only those tuples of "instructor X teaches" that pertain to instructors and the courses that they taught, we write:

  $\sigma_{instructor.id = teaches.id}$ (*instructor x teaches*)

Theta Join Operation

AALBORG UNIVERSITY

- The table corresponding to:

$$\sigma_{instructor.id = teaches.id}(instructor \times teaches)$$

instructor.ID	*name*	*dept_name*	*salary*	*teaches.ID*	*course_id*	*sec_id*	*semester*	*year*
10101	Srinivasan	Comp. Sci.	65000	10101	CS-101	1	Fall	2017
10101	Srinivasan	Comp. Sci.	65000	10101	CS-315	1	Spring	2018
10101	Srinivasan	Comp. Sci.	65000	10101	CS-347	1	Fall	2017
12121	Wu	Finance	90000	12121	FIN-201	1	Spring	2018
15151	Mozart	Music	40000	15151	MU-199	1	Spring	2018
22222	Einstein	Physics	95000	22222	PHY-101	1	Fall	2017
32343	El Said	History	60000	32343	HIS-351	1	Spring	2018
45565	Katz	Comp. Sci.	75000	45565	CS-101	1	Spring	2018
45565	Katz	Comp. Sci.	75000	45565	CS-319	1	Spring	2018
76766	Crick	Biology	72000	76766	BIO-101	1	Summer	2017
76766	Crick	Biology	72000	76766	BIO-301	1	Summer	2018
83821	Brandt	Comp. Sci.	92000	83821	CS-190	1	Spring	2017
83821	Brandt	Comp. Sci.	92000	83821	CS-190	2	Spring	2017
83821	Brandt	Comp. Sci.	92000	83821	CS-319	2	Spring	2018
98345	Kim	Elec. Eng.	80000	98345	EE-181	1	Spring	2017

Theta Join Operation

AALBORG UNIVERSITY

- The **join** operation allows us to combine a select operation and a Cartesian-Product operation into a single operation.
- Consider relations *r*(*R*) and *s*(*S*)
- Let "theta" be a predicate on attributes in the schema R "union" S. The join operation r $\bowtie_\theta$ s is defined as follows:

$$r \bowtie_\theta s = \sigma_\theta (r \times s)$$

- Thus

$$\sigma_{instructor.id = teaches.id} (instructor \times teaches)$$

- Can equivalently be written as

$$instructor \bowtie_{instructor.id = teaches.id} teaches$$

Theta Join Operation

AALBORG UNIVERSITY

- Example: Find the students whose score in quiz2 is higher than quiz1
- $quiz1 \bowtie_{quiz1.name = quiz2.name \wedge quiz2.score > quiz1.score} quiz\ 2$

quiz1

name	*score*
Anna	80
Bob	70
Cathy	90
David	100

quiz2

name	*score*
Anna	90
Bob	60
Cathy	100
David	80

Result

quiz1.name	*quiz1.score*	*quiz2.name*	*quiz2.score*
Anna	80	Anna	90
Cathy	90	Cathy	100

Theta Join Operation

AALBORG UNIVERSITY

- Example: For those students who have made donations, find their names, department name, and amounts of their donations
- $student \bowtie_{student.name = donation.name} donation$

student

name	*dept_name*
Anna	Comp. Sci.
Bob	Physics
Cathy	Music
David	Physics

donation

name	*amount*
Alice	100
Anna	150
Christian	120
David	180

Result

student.name	*dept_name*	*donation.name*	*amount*
Anna	Comp. Sci.	Anna	150
David	Physics	David	180

Any Problem

Theta Join Operation

AALBORG UNIVERSITY

- Use projection operation
- $\Pi_{student.name,\ dept\_name,\ amount}\ (student \bowtie_{student.name\ =\ donation.name} donation)$

student

name	*dept_name*
Anna	Comp. Sci.
Bob	Physics
Cathy	Music
David	Physics

donation

name	*amount*
Alice	100
Anna	150
Christian	120
David	180

Result

student.name	*dept_name*	*amount*
Anna	Comp. Sci.	150
David	Physics	180

Natural Join Operation

AALBORG UNIVERSITY

- Note 1: The join is performed based on the common attributes of the two relations
- Note 2: Each common attribute appears only once in the result
- *student* ⋈ *donation*

student

name	*dept_name*
Anna	Comp. Sci.
Bob	Physics
Cathy	Music
David	Physics

donation

name	*amount*
Alice	100
Anna	150
Christian	120
David	180

Result

name	*dept_name*	*amount*
Anna	Comp. Sci.	150
David	Physics	180

Natural Join Operation

- $(\sigma_{dept\_name = \text{“Physics”}} student) \bowtie donation$

student

name	*dept_name*
Anna	Comp. Sci.
Bob	Physics
Cathy	Music
David	Physics

donation

name	*amount*
Alice	100
Anna	150
Christian	120
David	180

Result

name	*dept_name*	*amount*
David	Physics	180

Exercise 2

AALBORG UNIVERSITY

- Write the following operations in relational algebra and think about the results
  - Cartesian-product operation
  - Theta join operation
    - *student.stu_ID = course.stu_ID*
  - Natural join operation

student

stu_ID	*stu_name*	*age*
1	John	20
2	Sarah	19
3	Mike	22

course

cou_ID	*cou_name*	*stu_ID*
101	*Math*	*1*
102	*History*	*3*
103	*Physics*	*2*

Exercise 2--Solution

AALBORG UNIVERSITY

- Cartesian-product operation
- *student* X *course*

student.stu_ID	*stu_name*	*age*	*cou_ID*	*cou_name*	*course.stu_ID*
1	John	20	*101*	*Math*	*1*
1	John	20	*102*	*History*	*3*
1	John	20	*103*	*Physics*	*2*
2	Sarah	19	*101*	*Math*	*1*
2	Sarah	19	*102*	*History*	*3*
2	Sarah	19	*103*	*Physics*	*2*
3	Mike	22	*101*	*Math*	*1*
3	Mike	22	*102*	*History*	*3*
3	Mike	22	*103*	*Physics*	*2*

Exercise 2--Solution

AALBORG UNIVERSITY

- Theta join operation $student \bowtie_{student.stu\_ID = course.stu\_ID} course$
  - *student.stu_ID = course.stu_ID*

student.stu_ID	*stu_name*	*age*	*cou_ID*	*cou_name*	*course.stu_ID*
1	John	20	*101*	*Math*	*1*
2	Sarah	19	*103*	*Physics*	*2*
3	Mike	22	*102*	*History*	*3*

- Natural join operation $student \bowtie course$

stu_ID	*stu_name*	*age*	*cou_ID*	*cou_name*
1	John	20	*101*	*Math*
2	Sarah	19	*103*	*Physics*
3	Mike	22	*102*	*History*

Union Operation

AALBORG UNIVERSITY

- The union operation allows us to combine two relations
- Notation: $r \cup s$
- For $r \cup s$ to be valid.
  - 1. *r*, *s* must have the same **arity** (same number of attributes)
  - 2. The attribute domains must be **compatible** (example: 2nd column of *r* deals with the same type of values as does the 2nd column of s)

Union Operation

AALBORG UNIVERSITY

- Example: find the ids of all courses taught in the Fall 2017 semester, or in the Spring 2018 semester, or in both
- $\Pi_{course\_id}(\sigma_{semester=\text{“Fall”} \wedge year=2017}(section)) \cup$
  $\Pi_{course\_id}(\sigma_{semester=\text{“Spring”} \wedge year=2018}(section))$

course_id	*sec_id*	*semester*	*year*	*building*	*room_number*	*time_slot_id*
BIO-101	1	Summer	2017	Painter	514	B
BIO-301	1	Summer	2018	Painter	514	A
CS-101	1	Fall	2017	Packard	101	H
CS-101	1	Spring	2018	Packard	101	F
CS-190	1	Spring	2017	Taylor	3128	E
CS-190	2	Spring	2017	Taylor	3128	A
CS-315	1	Spring	2018	Watson	120	D
CS-319	1	Spring	2018	Watson	100	B
CS-319	2	Spring	2018	Taylor	3128	C
CS-347	1	Fall	2017	Taylor	3128	A
EE-181	1	Spring	2017	Taylor	3128	C
FIN-201	1	Spring	2018	Packard	101	B
HIS-351	1	Spring	2018	Painter	514	C
MU-199	1	Spring	2018	Packard	101	D
PHY-101	1	Fall	2017	Watson	100	A

section

Result

course_id
CS-101
CS-315
CS-319
CS-347
FIN-201
HIS-351
MU-199
PHY-101

Union Operation

- Example: find the persons who are either students or volunteers
- *student* ∪ *volunteer*

student

name	*age*
Anna	19
Bob	22
Cathy	20
David	21

volunteer

name	*age*
Alice	23
Anna	19
Christian	24
David	21

Result

name	*age*
Alice	23
Anna	19
Bob	22
Cathy	20
Christian	24
David	21

Union Operation

- Example: find the names of the persons who are either students or volunteers
- $\prod_{name}(student \cup volunteer)$
- $\prod_{name}(student) \cup \prod_{name}(volunteer)$

student

name	*age*
Anna	19
Bob	22
Cathy	20
David	21

volunteer

name	*age*
Alice	23
Anna	19
Christian	24
David	21

Result

name
Alice
Anna
Bob
Cathy
Christian
David

Union Operation

AALBORG UNIVERSITY

- Example: find the persons who are either students or volunteers
- $student \cup volunteer$ **wrong!**
- $\prod_{name}(student) \cup volunteer$ **correct version**

student

name	*age*
Anna	19
Bob	22
Cathy	20
David	21

volunteer

name
Alice
Anna
Christian
David

Result

name
Alice
Anna
Bob
Cathy
Christian
David

Set-Intersection Operation

AALBORG UNIVERSITY

- The set-intersection operation allows us to find tuples that are in both the input relations.
- Notation: $r \cap s$
- Assume:
  - *r*, *s* have the same arity
  - attributes of *r* and *s* are compatible

Set-Intersection Operation

AALBORG UNIVERSITY

- Example: find the ids of all courses taught in both the Fall 2017 and the Spring 2018 semesters
- $\Pi_{course\_id}(\sigma_{semester=\text{“Fall”} \wedge year=2017}(section)) \cap$
  $\Pi_{course\_id}(\sigma_{semester=\text{“Spring”} \wedge year=2018}(section))$

course_id	*sec_id*	*semester*	*year*	*building*	*room_number*	*time_slot_id*
BIO-101	1	Summer	2017	Painter	514	B
BIO-301	1	Summer	2018	Painter	514	A
CS-101	1	Fall	2017	Packard	101	H
CS-101	1	Spring	2018	Packard	101	F
CS-190	1	Spring	2017	Taylor	3128	E
CS-190	2	Spring	2017	Taylor	3128	A
CS-315	1	Spring	2018	Watson	120	D
CS-319	1	Spring	2018	Watson	100	B
CS-319	2	Spring	2018	Taylor	3128	C
CS-347	1	Fall	2017	Taylor	3128	A
EE-181	1	Spring	2017	Taylor	3128	C
FIN-201	1	Spring	2018	Packard	101	B
HIS-351	1	Spring	2018	Painter	514	C
MU-199	1	Spring	2018	Packard	101	D
PHY-101	1	Fall	2017	Watson	100	A

section

Result

course_id
CS-101

Set-Intersection Operation

AALBORG UNIVERSITY

- Example: find the persons who are both students and volunteers
- *student* ∩ *volunteer*

student

name	*age*
Anna	19
Bob	22
Cathy	20
David	21

volunteer

name	*age*
Alice	23
Anna	19
Christian	24
David	21

Result

name	*age*
Anna	19
David	21

Set-Intersection Operation

AALBORG UNIVERSITY

- Example: find the names of the persons who are both students and volunteers
- $\prod_{name}(student \cap volunteer)$
- $\prod_{name}(student) \cap \prod_{name}(volunteer)$

student

name	*age*
Anna	19
Bob	22
Cathy	20
David	21

volunteer

name	*age*
Alice	23
Anna	19
Christian	24
David	21

Result

name
Anna
David

Set-Intersection Operation

AALBORG UNIVERSITY

- Example: find the persons who are both students and volunteers
- *student* ∩ *volunteer* **wrong!**
- Π_{name} (*student*) ∩ *volunteer* **correct version**

student

name	*age*
Anna	19
Bob	22
Cathy	20
David	21

volunteer

name
Alice
Anna
Christian
David

Result

name
Anna
David

Set-Difference Operation

AALBORG UNIVERSITY

- The set-difference operation allows us to find tuples that are in one relation but are not in another.
- Notation $r - s$
- Set differences must be taken between compatible relations.
  - r and s must have the same arity
  - attribute domains of r and s must be compatible

Set-Difference Operation

AALBORG UNIVERSITY

- Example: find the ids of all courses taught in the Fall 2017 semester, but not in the Spring 2018 semester
- $\Pi_{course\_id}(\sigma_{semester=\text{“Fall”} \wedge year=2017}(section)) -$
  $\Pi_{course\_id}(\sigma_{semester=\text{“Spring”} \wedge year=2018}(section))$

course_id	*sec_id*	*semester*	*year*	*building*	*room_number*	*time_slot_id*
BIO-101	1	Summer	2017	Painter	514	B
BIO-301	1	Summer	2018	Painter	514	A
CS-101	1	Fall	2017	Packard	101	H
CS-101	1	Spring	2018	Packard	101	F
CS-190	1	Spring	2017	Taylor	3128	E
CS-190	2	Spring	2017	Taylor	3128	A
CS-315	1	Spring	2018	Watson	120	D
CS-319	1	Spring	2018	Watson	100	B
CS-319	2	Spring	2018	Taylor	3128	C
CS-347	1	Fall	2017	Taylor	3128	A
EE-181	1	Spring	2017	Taylor	3128	C
FIN-201	1	Spring	2018	Packard	101	B
HIS-351	1	Spring	2018	Painter	514	C
MU-199	1	Spring	2018	Packard	101	D
PHY-101	1	Fall	2017	Watson	100	A

section

Result

course_id
CS-347
PHY-101

Exercise 3

- Write the following operations in relational algebra and think about the results
  - Union operation
  - Set-intersection operation
  - Set-difference operation (in *Course_A*, not in *Course_B*)

Course_A

cou_ID	*cou_name*
101	Math
102	History
103	Physics

Course_B

cou_ID	*cou_name*
102	History
103	Physics
104	Chemistry

Exercise 3--Solution

AALBORG UNIVERSITY

- Union
- *Course_A* ∪ *Course_B*

cou_ID	*cou_name*
101	Math
102	History
103	Physics
104	Chemistry

- Set-intersection
- *Course_A* ∩ *Course_B*

cou_ID	*cou_name*
102	History
103	Physics

- Set-difference
  - (in *Course_A*, not in *Course_B*)
- *Course_A* – *Course_B*

cou_ID	*cou_name*
101	Math

The Assignment Operation

AALBORG UNIVERSITY

- It is convenient at times to write a relational-algebra expression by assigning parts of it to temporary relation variables.
- The assignment operation is denoted by $\leftarrow$ and works like assignment in a programming language.
- Example: Find all instructors in the Physics or Music department.

$$Physics \leftarrow \sigma_{dept\_name= \text{“Physics”}} (instructor)$$
$$Music \leftarrow \sigma_{dept\_name= \text{“Music”}} (instructor)$$
$$Physics \cup Music$$

- It can also be used to attributes
  - $\prod_{A \leftarrow B}(r)$

The Assignment Operation

AALBORG UNIVERSITY

- Example: find all instructors in the Physics or Music departments

  $Physics \leftarrow \sigma_{dept\_name= \text{“Physics”}} (instructor)$

  $Music \leftarrow \sigma_{dept\_name= \text{“Music”}} (instructor)$

  $Physics \cup Music$

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

instructor

Physics

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
33456	Gold	Physics	87000

Music

ID	*name*	*dept_name*	*salary*
15151	Mozart	Music	40000

Result

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
33456	Gold	Physics	87000
15151	Mozart	Music	40000

The Rename Operation

AALBORG UNIVERSITY

- The results of relational-algebra expressions do not have a name that we can use to refer to them. The rename operator, ρ, is provided for that purpose
- The expression:

  $\rho_x(E)$

  returns the result of expression *E* under the name *x*
- Another form of the rename operation:

  $\rho_{x(A1,A2,..An)}(E)$

The Rename Operation

AALBORG UNIVERSITY

- Example: find all instructor in the Physics or Music department

$\rho_{Physics}(\sigma_{dept\_name=\text{“Physics”}}(instructor))$ $Physics \leftarrow \sigma_{dept\_name=\text{“Physics”}}(instructor)$

$\rho_{Music}(\sigma_{dept\_name=\text{“Music”}}(instructor))$ $Music \leftarrow \sigma_{dept\_name=\text{“Music”}}(instructor)$

$Physics \cup Music$

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

instructor

Physics

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
33456	Gold	Physics	87000

Music

ID	*name*	*dept_name*	*salary*
15151	Mozart	Music	40000

Result

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
33456	Gold	Physics	87000
15151	Mozart	Music	40000

The Rename Operation

AALBORG UNIVERSITY

$\rho_{Physics\ (p.ID,\ p.name,\ p.dept,\ p.salary)}\ (\sigma_{dept\_name=\text{“}Physics\text{”}}\ (instructor))$

$\rho_{Music\ (m.ID,\ m.name,\ m,dept,\ m.salary)}\ (\sigma_{dept\_name=\text{“}Music\text{”}}\ (instructor))$

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

instructor

Physics

p.ID	*p.name*	*p.dept*	*p.salary*
22222	Einstein	Physics	95000
33456	Gold	Physics	87000

Music

m.ID	*m.name*	*m.dept*	*m.salary*
15151	Mozart	Music	40000

Left Join Operation

AALBORG UNIVERSITY

- Example: For each student, find the amount of his/her donation
- *Student* ⟕ *donation*

student

name	*dept_name*
Anna	Comp. Sci.
Bob	Physics
Cathy	Music
David	Physics

donation

name	*amount*
Alice	100
Anna	150
Christian	120
David	180

Result

name	*dept_name*	*amount*
Anna	Comp. Sci.	150
Bob	Physics	null
Cathy	Music	null
David	Physics	180

Right Join Operation

AALBORG UNIVERSITY

- Example: For each donator, find the department he/she is in
- *Student* ⟖ *donation*

student

name	*dept_name*
Anna	Comp. Sci.
Bob	Physics
Cathy	Music
David	Physics

donation

name	*amount*
Alice	100
Anna	150
Christian	120
David	180

Result

name	*dept_name*	*amount*
Alice	null	100
Anna	Comp. Sci.	150
Christian	null	120
David	Physics	180

Outer Join Operation

- *Student* ⟗ *donation*

student

name	*dept_name*
Anna	Comp. Sci.
Bob	Physics
Cathy	Music
David	Physics

donation

name	*amount*
Alice	100
Anna	150
Christian	120
David	180

Result

name	*dept_name*	*amount*
Alice	null	100
Anna	Comp. Sci.	150
Bob	Physics	null
Cathy	Music	null
Christian	null	120
David	Physics	180

Grouping and Aggregation

AALBORG UNIVERSITY

- Tuples with the same attribute values (for a specified list of attributes) are grouped.
- An aggregate function is applied to each group (computing one value for each group).
- Typical aggregate functions
  - count: number of tuples in a group
  - sum: sum of attribute values in a group
  - min, max, avg: minimum, maximum, average of attribute values in a group
- Notation: $\gamma_{L;\,F}(r)$
  - L list of attributes for grouping
  - F aggregate function

Grouping and Aggregation

AALBORG UNIVERSITY

- Example: determine the number of students per semester
- $\gamma_{semester;\ count(*)}(student)$

student

name	*semester*	*age*
Alice	7	24
Anna	3	20
Bob	3	19
Cathy	5	22
Christian	5	20
David	1	18

Result

semester	*count(*)*
7	1
3	2
5	2
1	1

Grouping and Aggregation

AALBORG UNIVERSITY

- Example: determine the number of students per semester and the minimum age in each group
- $\gamma_{semester;\ count(*),\ min(age)}\ (student)$

student

name	*semester*	*age*
Alice	7	24
Anna	3	20
Bob	3	19
Cathy	5	22
Christian	5	20
David	1	18

Result

semester	*count(*)*	*min(age)*
7	1	24
3	2	19
5	2	20
1	1	18

Equivalent Queries

AALBORG UNIVERSITY

- There is more than one way to write a query in relational algebra.
- Example: find instructors in the Physics department with salary greater than 90,000
- Query 1
  - $\sigma_{dept\_name=\text{“Physics”} \wedge salary > 90,000}(instructor)$
- Query 2
  - $\sigma_{dept\_name=\text{“Physics”}}(\sigma_{salary > 90.000}(instructor))$
- The two queries are not identical; they are, however, equivalent -- they give the same result on any database.

Summary

AALBORG UNIVERSITY

- Relational Model
  - Relation, tuple, attribute
  - Schema and instance
  - Keys: Superkey, candidate key, primary key, foreign key
- The Relational Algebra
  - select: σ
  - project: Π
  - Cartesian product: x
  - join: $\bowtie$
  - union: $\cup$
  - set-intersection: $\cap$
  - set-difference: –
  - rename: ρ

AALBORG UNIVERSITY

Next Lecture

- SQL
  - Explain and use the SQL data model
  - Create non-trivial database tables
  - Modify non-trivial database tables
  - Create non-trivial SQL statements