HALMSTAD UNIVERSITY

DS4001 Databases (7.5 credits)

Lecture 4 –SQL Statement & Exercises

Yuantao Fan

yuantao.fan@hh.se

Halmstad University

Overview

- Ordering
- Aliansing
- Aggregation
- Grouping
- Having
- Lab1 Introduction

Example Database

Course(cid, Course_name, Course_code, Credit_hours)

cid	course_name	course_code	credits
1	Intro to Computer Science	CS1310	4
2	Data Structures	CS3320	4
3	Discrete Mathematics	MATH2410	3
4	Database	CS3380	3

Teacher(tid, full_name, age, nationality)

tid	full_name	age	nationality
11	John Smith	42	America
22	Jens Jonathon	31	Sweden
33	Stefan Miller	39	Sweden
44	Kayle Persson	33	UK

Teaches(tid, cid, hours)

tid	cid	hours
11	1	80
11	2	100
22	4	50
33	4	50
44	3	100

Ordering

- Output control
- Ordering the output tuples by the values in one for more columns
- Syntax
  - ORDER BY <column> [ASC/DESC]

tid	full_name	age	nationality
11	John Smith	42	America
22	Jens Jonathon	31	Sweden
33	Stefan Miller	39	Sweden
44	Kayle Persson	33	UK

```
SELECT tid, full_name, age
FROM Teacher
WHERE nationality='SWEDEN'
ORDER BY age;
```

Ordering

- Output control
- Ordering the output tuples by the values in one for more columns
- Syntax
  - ORDER BY <column> [ASC/DESC]

tid	full_name	age	nationality
11	John Smith	42	America
22	Jens Jonathon	31	Sweden
33	Stefan Miller	39	Sweden
44	Kayle Persson	33	UK

```
SELECT tid, full_name, age
FROM Teacher
WHERE nationality='SWEDEN'
ORDER BY age;
```

tid	full_name	age
22	Jens Jonathon	31
33	Stefan Miller	39

Ordering

- Output control
- Ordering the output tuples by the values in one for more columns
- Syntax
  - ORDER BY <column> [ASC/DESC]

tid	full_name	age	nationality
11	John Smith	42	America
22	Jens Jonathon	31	Sweden
33	Stefan Miller	39	Sweden
44	Kayle Persson	33	UK

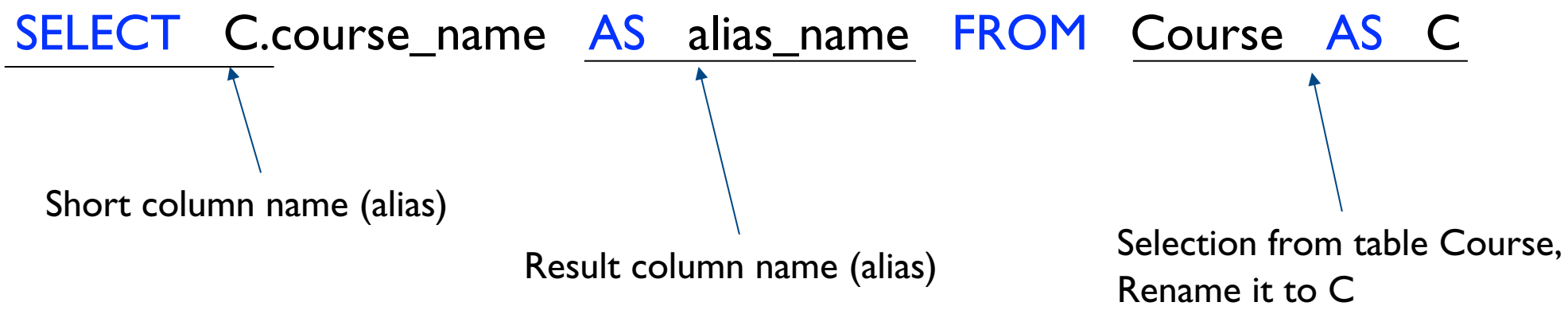
```
SELECT tid, full_name, age
FROM Teacher
WHERE nationality='SWEDEN'
ORDER BY age DESC;
```

tid	full_name	age
33	Stefan Miller	39
22	Jens Jonathon	31

What is the output with ‘ORDER BY 1 DESC’?

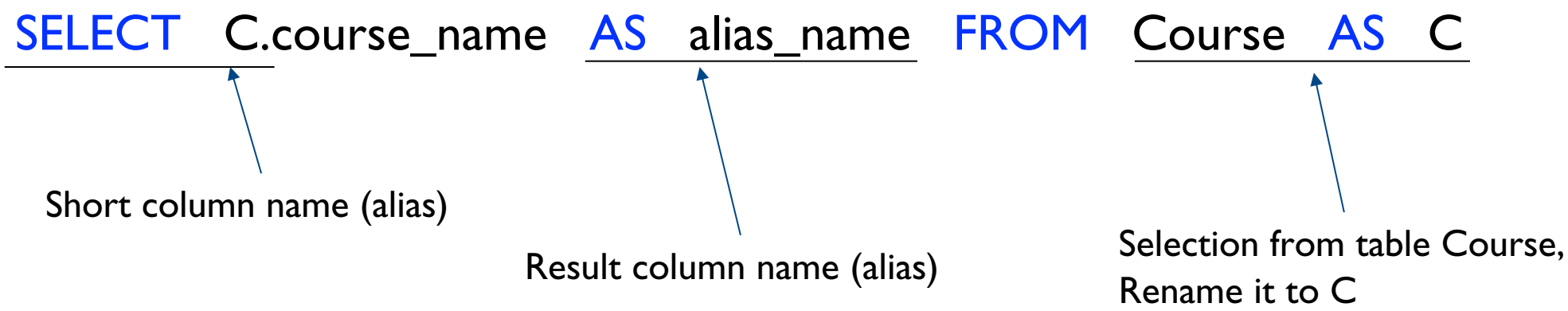
Aliasing for Tables and Columns

- Columns in SELECT and Tables in FROM can be renamed



Aliasing for Tables and Columns

- Columns in SELECT and Tables in FROM can be renamed

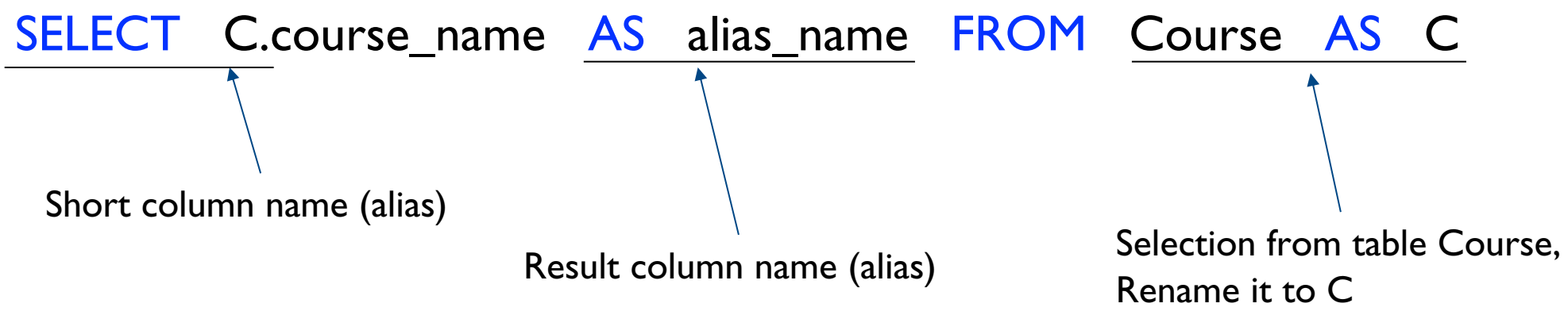


cid	course_name	course_code	credits
1	Intro to Computer Science	CS1310	4
2	Data Structure	CS3320	4
3	Discrete Mathematics	MATH2410	3
4	Database	CS1310	3

```
SELECT C.course_name AS alias_name
FROM Course AS C
```

Aliasing for Tables and Columns

- Columns in SELECT and Tables in FROM can be renamed



cid	course_name	course_code	credits
1	Intro to Computer Science	CS1310	4
2	Data Structure	CS3320	4
3	Discrete Mathematics	MATH2410	3
4	Database	CS1310	3

```
SELECT C.course_name AS alias_name
FROM Course AS C
```

alias_name
Intro to Computer Science
Data Structure
Discrete Mathematics
Database

Aliasing for Tables and Columns

- Columns in SELECT and Tables in FROM can be renamed

```
SELECT C.course_name AS alias_name FROM Course AS C
```

Short column name (alias)

Result column name (alias)

Selection from table Course,
Rename it to C

tid	full_name	age	nationality
11	John Smith	42	America
22	Jens Jonathon	31	Sweden
33	Stefan Miller	39	Sweden
44	Dona Kahill	33	UK

```
SELECT nationality AS COUNTRY
FROM Teacher
WHERE nationality='SWEDEN';
```

Aliasing for Tables and Columns

- Columns in SELECT and Tables in FROM can be renamed

```
SELECT C.course_name AS alias_name FROM Course AS C
```

Short column name (alias)

Result column name (alias)

Selection from table Course, Rename it to C

tid	full_name	age	nationality
11	John Smith	42	America
22	Jens Jonathon	31	Sweden
33	Stefan Miller	39	Sweden
44	Dona Kahill	33	UK

```
SELECT nationality AS COUNTRY
FROM Teacher
WHERE nationality='SWEDEN';
```

COUNTRY
Sweden
Sweden

Aggregation

- Aggregate function are built-in and can be applied in the SELECT output list

- COUNT(<column>) – nuber of values (or rows) from for column

- AVG(<column>) – compute the mean of all values in the column
- SUM(<column>) - acquire the sum of all values in the column

- MIN(<column>) – acquire the mininum value from the column
- MAX(<column>) - acquire the maximum value from the column

Aggregation

- Aggregate function are almost always used in the output from SELECT statement

- Acquire numbers of teachers from Sweden:

Aggregation

- Aggregate function are almost always used in the output from SELECT statement

- Acquire numbers of teachers from Sweden:

```
SELECT COUNT(nationality) AS cnt
FROM Teacher
WHERE nationality='SWEDEN';
```

tid	full_name	age	nationality
11	John Smith	42	America
22	Jens Jonathon	31	Sweden
33	Stefan Miller	39	Sweden
44	Dona Kahill	33	UK

```
SELECT COUNT(nationality) AS cnt
FROM Teacher
WHERE nationality='SWEDEN';
```

cnt
2

Aggregation

- Aggregate function are almost always used in the output from SELECT statement

- Acquire numbers of teachers from Sweden:

```
SELECT COUNT(nationality) AS cnt
FROM Tea
WHERE na
```

```
SELECT COUNT(*) AS cnt
FROM Teacher
WHERE nationality='SWEDEN';
```

tid	full_name	age	nationality
11	John Smith	42	America
22	Jens Jonathon	31	Sweden
33	Stefan Miller	39	Sweden
44	Dona Kahill	33	UK

```
SELECT COUNT(nationality) AS cnt
FROM Teacher
WHERE nationality='SWEDEN';
```

cnt
2

Aggregation

- Aggregate function are almost always used in the output from SELECT statement

- Acquire numbers of teachers from Sweden:

```
SELECT COUNT(nationality) AS cnt
FROM Tea
WHERE na
```

```
SELECT COUNT(*) AS cnt
FRO
WHE
```

```
SELECT COUNT(1) AS cnt
FROM Teacher
WHERE nationality='SWEDEN';
```

tid	full_name	age	nationality
11	John Smith	42	America
22	Jens Jonathon	31	Sweden
33	Stefan Miller	39	Sweden
44	Dona Kahill	33	UK

```
SELECT COUNT(nationality) AS cnt
FROM Teacher
WHERE nationality='SWEDEN';
```

cnt
2

Aggregation

- Aggregate function are almost always used in the output from SELECT statement

- Acquire the average age of teachers from Sweden:

```
SELECT AVG(age)
FROM Teacher
WHERE nationality='SWEDEN';
```

tid	full_name	age	nationality
11	John Smith	42	America
22	Jens Jonathon	31	Sweden
33	Stefan Miller	39	Sweden
44	Dona Kahill	33	UK

```
SELECT AVG(age)
FROM Teacher
WHERE nationality='SWEDEN';
```

AVG(age)
35.0000

Aggregation with multiple columns

- Acquire the number of teachers and their average age that come from Sweden

Aggregation with multiple columns

- Acquire the number of teachers and their average age that come from Sweden

```
SELECT COUNT(tid), VG(age)
FROM Teacher
WHERE nationality='SWEDEN';
```

tid	full_name	age	nationality
11	John Smith	42	America
22	Jens Jonathon	31	Sweden
33	Stefan Miller	39	Sweden
44	Dona Kahill	33	UK

```
SELECT COUNT(tid), AVG(age)
FROM Teacher
WHERE nationality='SWEDEN';
```

COUNT(tid)	AVG(age)
2	35.0000

Aggregation

- DISTINCT
  - Acquire distinctive values from column(s)
  - Operation set

- Acquire the nationalities of the teachers in the table (no repeat)

Aggregation

- DISTINCT
  - Acquire distinctive values from column(s)
  - Operation set

- Acquire the nationalities of the teachers in the table (no repeat)

```
SELECT DISTINCT(nationality)
FROM Teacher
```

nationality
America
Sweden
UK

Aggregation

- DISTINCT
  - Acquire distinctive values from column(s)
  - Operation set

- COUNT, SUM, AVG support DISTINCT
  - count numbers of the unique nationalities of all teachers in the table

Aggregation

- DISTINCT
  - Acquire distinctive values from column(s)
  - Operation set

- COUNT, SUM, AVG support DISTINCT
  - count numbers of the unique nationalities of all teachers in the table

```
SELECT COUNT(DISTINCT(nationality))
FROM Teacher
```

COUNT(DISTINCT(nationality))
3

Aggregation

- Note that output of other columns excluded from the aggregation is undefined
- Compute the average age of teachers in each course

Aggregation

- Note that output of other columns excluded from the aggregation is undefined

- Compute the average age of teachers in each course

?
```
SELECT AVG(t.age), tt.cid
FROM Teacher AS t, Teaches AS tt
WHERE t.tid=tt.tid
```

Aggregation

- Note that output of other columns excluded from the aggregation is undefined

- Compute the average age of teachers in each course

```
SELECT AVG(t.age), tt.cid
FROM Teacher AS t, Teaches AS tt
WHERE t.tid=tt.tid
```

Aggregation

- Note that output of other columns excluded from the aggregation is undefined
- Compute the average age of teachers in each course
- Requires group by

```
SELECT AVG(t.age), tt.cid
FROM Teacher AS t, Teaches AS tt
WHERE t.tid=tt.tid
GROUP BY tt.cid
```

AVG(t.age)	cid
42.0000	1
42.0000	2
32.0000	3
39.0000	4

Teacher

tid	full_name	age	nationality
11	John Smith	42	America
22	Jens Jonathon	31	Sweden
33	Stefan Miller	39	Sweden
44	Dona Kahill	33	UK

Teaches

tid	cid	hours
11	1	50
11	2	100
22	3	80
33	4	80
44	3	100

Aggregation

- Note that output of other columns excluded from the aggregation is undefined
- Compute the average age of teachers in each course
- Requires group by

```
SELECT AVG(t.age), tt.cid, cc.course_name
FROM Teacher AS t, Teaches AS tt, Course AS cc
WHERE t.tid=tt.tid  AND tt.cid=cc.cid
GROUP BY tt.cid
```

AVG(t.age)	cid	course_name
42.0000	1	Intro to Computer Science
42.0000	2	Data Structure
32.0000	3	Discrete Mathematics
39.0000	4	Database

Course

cid	course_name	course_code	credits
1	Intro to Computer Science	CS1310	4
2	Data Structure	CS3320	4
3	Discrete Mathematics	MATH2410	3
4	Database	CS1310	3

Teacher

tid	full_name	age	nationality
11	John Smith	42	America
22	Jens Jonathon	31	Sweden
33	Stefan Miller	39	Sweden
44	Dona Kahill	33	UK

Teaches

tid	cid	hours
11	1	50
11	2	100
22	3	80
33	4	80
44	3	100

Aggregation

- Note that output of other columns excluded from the aggregation is undefined
- Compute the average age of teachers in each course
- Non-aggregated values in SELECT output clause must appear in GROUP BY clused

```
SELECT AVG(t.age), tt.cid, t.full_name
FROM Teacher AS t, Teaches AS tt
WHERE t.tid=tt.tid
GROUP BY tt.cid
```

Aggregation

- Note that output of other columns excluded from the aggregation is undefined
- Compute the average age of teachers in each course
- Non-aggregated values in SELECT output clause must appear in GROUP BY clused

```
SELECT AVG(t.age), tt.cid, t.full_name
FROM Teacher AS t, Teaches AS tt
WHERE t.tid=tt.tid
GROUP BY tt.cid, t.full_name;
```

AVG(t.age)	cid	full_name
42.0000	1	John Smith
42.0000	2	John Smith
31.0000	3	Jens Jonathon
39.0000	4	Stefan Miller
33.0000	3	Dona Kahill

Aggregation

- Note that output of other columns excluded from the aggregation is undefined
- Compute the average age of teachers in each course
- What if we would like to acquire courses taught by young teachers?

AVG(t.age)	cid	course_name
42.0000	1	Intro to Computer Science
42.0000	2	Data Structure
32.0000	3	Discrete Mathematics
39.0000	4	Database

Aggregation

- Note that output of other columns excluded from the aggregation is undefined
- Compute the average age of teachers in each course
- What if we would like to acquire courses taught by young teachers?

```
SELECT AVG(t.age) AS avg_age, tt.cid, cc.course_name
FROM Teacher AS t, Teaches AS tt, Course AS cc
WHERE t.tid=tt.tid AND tt.cid=cc.cid AND avg_age < 35
GROUP BY tt.cid;
```

avg_age	cid	course_name
42.0000	1	Intro to Computer Science
42.0000	2	Data Structure
32.0000	3	Discrete Mathematics
39.0000	4	Database

Aggregation

- HAVING
  - Filter based on aggregated results
  - Consider it a WHERE clause for GROUP BY

```
SELECT AVG(t.age) AS avg_age, tt.cid, cc.course_name
FROM Teacher AS t, Teaches AS tt, Course AS cc
WHERE t.tid=tt.tid AND tt.cid=cc.cid
GROUP BY tt.cid
HAVING avg_age < 35;
```

avg_age	cid	course_name
32.0000	3	Discrete Mathematics

avg_age	cid	course_name
42.0000	1	Intro to Computer Science
42.0000	2	Data Structure
32.0000	3	Discrete Mathematics
39.0000	4	Database

Lab 1 Introduction

- Objective & learning outcome
  - Learn how to use DDL and DML to create tables, manipulate the content in the table and writting queries
  - Get familiar with basic SQL statement

- Content
  - Create a relational database with multiple tables
  - writting queries given a question/description
    - Practising basic SQL statement
  - Propose queries
  - Get familiar with a real world database - imdb

Lab 1 Introduction

- Part 1 – create and work with a database example
  - Create a database
  - Practise SQL statements
- Part 2 – working with an existing database
  - Create databases with the given script
  - Practise SQL statements
- Part 3 – exploring a real-world dataset
  - Exploring a dataset with queries

Lab 1.1 Create and working with a database example

Student

sid	full_name	major	age	GPA
c1	Alice	CS	21	4.0
p2	Albert	PHY	22	3.9
e3	Tim	EE	20	3.9
m4	Kayle	MATH	19	3.8
p5	Yasuo	PHY	19	3.7

Course

cid	course_name	course_code	credits
11	Linear algebra	MATH105	5
22	Algorithms	CS101	5
33	Databases	DS001	4.5
44	Physics I	PHY001	6

Enrolled

sid	cid	grade
c1	11	A
c1	33	A
p2	44	A
p5	44	B
m4	11	A
p2	11	B
m4	22	B
p5	33	C
c1	22	A

Note that sid and cid in Encolled are forign keys
Refering to Student(sid) and Course(cid)

Lab 1.1 Tasks

- Using DDL and DML to create a database
  - Write SQL code
  - Test your script and verify the output databse
  - Save and name your script as “create_db_sqg.sql”

- Queries
  - Select all students above the age of 20
  - Who is the oldest student?
  - Count the number of students with age below 20
  - How many types of majors were these students admitted to?
  - What is the average GPA of students with age above 20?
  - What is the average GPA of students studying the Physics major?
  - What is the average age of students who took Linear algebra courses?
  - How many courses has Alice registered for?
  - How many credits has Alice registered?
  - How many credits have students with age below 20 registered to?

- Propose 2 or more queries of practical usage

Lab 1.2 Tasks

- Download “example-create-databases.sql” from blackboard
  - Execute the script and take a look at the created databases

- Queries (see next slide)
  - sql_Inventory
    - Products
  - sql_HR
    - Employees and offices
  - sql_Invoicing
    - Clients, invoices, payment_methods, and payments
  - sql_store
    - Customers, products, orders, …

- For each database
  - propose 2 or more queries of practical usage

Lab 1.2 Queries

- sql_inventory
  - What is the most valuable asset in the inventory?
  - How much does the entire inventory worth?
- sql_hr
  - Where is the largest office (in terms of numbers of employees) located?
  - Who sits alone?
- sql_invoicing
  - What is the most common payment method?
  - Which client seems to be the most important one? Motivate your approach and answer.
- sql_store
  - How much do order 2 worth?
  - Which customer has their order delivered?

Lab 1.3 Tasks

- Tool
  - SQLite3 – make sure you have access to it
  - https://www.sqlite.org/download.html
- Movie database
  - Download the zip file of the database from the blackboard
- Queries
  - How many movies have the highest rating?
  - What are the most common genres in this database?
  - Which movie is the longest?
- Propose 2 or more queries of your interest