Relational Database Introduction

UBCO Master of Data Science – DATA 540

Introductions

Instructor: Dr. Ifeoma Adaji

- Assistant Professor, Computer Science since July 2021
- Ph.D., Computer Science, University of Saskatchewan, Canada, 2020
- M.Sc., Computer Science, University of Aberdeen, Scotland, UK
- Research area: Data science, social network analysis, user modeling, machine learning
- Industry experience: Data scientist at Insightrix Research

TA: A.K.M. Amanat Ullah

- Ph.D. in Computer Science student

The Essence of the Course

The overall goal of this course is for you to:

Understand how to create, query, and program with databases to perform data analysis with larger data sets.

This course covers database techniques and software including relational and NoSQL databases, XML, and JSON.

Data analysts must have fluency in SQL and other query languages to extract and analyze data from a variety of systems.

My Course Goals

1) Provide the information in a simple, concise, and effective way for learning.
2) Strive for ***all*** students to understand the material and excel at the course.
3) Be available for questions during class time, office hours, and at other times as needed.
4) Teach students how to be a sophisticated database user (by being fluent in SQL), write programs that access a database, and be able to read and understand database design (ER) diagrams.

Course Objectives

1) Understand the use case for databases and the relational model for data storage.
2) Fluency in SQL including SQL DDL (`CREATE`, `DROP`, `INSERT`, `UPDATE`, `DELETE`) and SQL queries using `SELECT`.
3) Construct programs that access a database to read data, perform analysis, and output results.
4) Exposure to database technologies like NoSQL, JSON, and XML.
5) Understand how to read database design (ER) diagrams and convert to the relational model.

Academic Dishonesty

Cheating is strictly prohibited and is taken very seriously by UBC.

A guideline to what constitutes cheating:

- Labs
  - Submitting code produced by others.
  - Working in groups to solve questions and/or comparing answers to questions once they have been solved (except for group assignments).
  - Discussing HOW to solve a particular question instead of WHAT the question involves.
- Exams
  - Only materials permitted by instructor should be in the exam.

Academic dishonesty may result in a "F" for the course and removal from the MDS program.

How to Excel in This Course

Attend ***every*** class:

- Read notes ***before*** class as preparation and try the questions.
- Participate in class exercises and questions.

Attend and complete all labs:

- Labs practice the fundamental employable skills as well as being for marks.

Practice on your own. Practice makes ~~perfect~~ improvement.

- Do more questions than in the labs.
- Read the additional reference material and perform practice questions.

Systems and Tools

Course material is on GitHub.

Marks are distributed on Canvas.

Your laptop will be used to install all software and run programs.

The Lab Assignments

Weekly lab assignments are worth ***30%*** of your overall grade.

Lab assignments may take more than the two hours lab time.

You have until the week after the lab to complete it.

- No late labs will be accepted.
- A lab may be handed in any time before the due date and may be marked immediately by the TA in the lab.

Lab assignments are done individually.

The lab assignments are critical to learning the material and are designed both to prepare you for the quizzes and build up your skills!

The In-Class Clicker Quizzes

To promote understanding, 10% of your overall grade is allocated to answering in-class questions.

These questions are answered electronically via iClicker.

- Search for Data 540-2023 on iClicker

There will be about 50 questions throughout the semester. Each question is worth 1 mark, and you need at least 40 right answers to get the full 5%.

  - That is, if you answer 30 questions right, you get 30/40 or 75%. Thus, do not worry if you forget your clicker one day!

Database Survey Question

Question: Have you used any of these database systems?

A) MySQL

B) Microsoft Access or SQL Server

C) PostgreSQL

D) Used more than two different databases

E) Used no databases

What Grade are You Expecting to Get?

A) A

B) B

C) C

D) D

F) F

Why This Course is Important

Database systems store the majority of data that data analysts use.

Key skills:

- ***Query/update databases*** – SQL to extract information from existing databases.
- ***Program with databases*** – Learn how to use Python and R to access data in a database then perform further analysis.
- ***Be aware of data technologies*** – XML, NoSQL, JSON, and a variety of others.

Objectives of Today's Class

- Define: database, DBMS, schema, metadata
- Define program-data independence/data abstraction and explain how it is achieved by databases but not by file systems.
- Define: relation, attribute, tuple, domain, degree, cardinality
- Define DDL and DML. What is the difference?
- List the properties of relations.
- Define: superkey, key, candidate key, primary key, foreign key
- Define: integrity, constraints, domain constraint, entity integrity constraint, referential integrity constraint
- Given a relation be able to:
  - identify its cardinality, degree, domains, keys, and superkeys
  - determine if constraints are being violated
- Define: relational algebra, query language

Why Relational Databases?

Relational databases allow for the storage and analysis of large amounts of data.

Relational databases are the most common form of database used by companies and organizations for data management.

Since a significant amount of data is stored in relational databases, understanding how to create and query these databases using the SQL standard is a very valuable skill.

What is a database?

A ***database*** is a collection of logically related data for a particular domain.

A ***database management system*** (***DBMS***) is software designed for the creation and management of databases.

- e.g. Oracle, DB2, Microsoft Access, MySQL, SQL Server, MongoDB

Bottom line: A ***database*** is the *data* stored and a ***database system*** is the *software* that manages the data.

Databases in the Real-World

Databases are everywhere in the real-world even though you do not often interact with them directly.

- $40 billion dollar annual industry

Examples:

- Retailers manage their products and sales using a database.
  - Wal-Mart has one of the largest databases in the world!
- Online web sites such as Amazon, eBay, and Expedia track orders, shipments, and customers using databases.
- The university maintains all your registration information and marks in a database that is accessible over the Internet.

Can you think of other examples?

What data do you have?

Data Independence and Abstraction

Without a database, applications use files to store data *persistently*. A ***file-based system*** has several problems: code and data duplication, high maintenance costs, and difficulty in supporting multiple users.

- There is no ***program-data independence*** separating the application from the data it is manipulating. If the data file changes, the code must be changed.

Databases provide ***data abstraction*** allowing the internal representation of the data to change without affecting programs that use the object through an external definition.

- The DBMS takes the description of the data and handles the low-level details of how to store it, retrieve it, and provide concurrent access to it.

Database System Properties

A database system provides *efficient*, *convenient*, and *safe multi-user* storage and access to *massive* amounts of *persistent* data.

Efficient - Able to handle large data sets and complex queries without searching all files and data items.

Convenient - Easy to write queries to retrieve data.

Safe - Protects data from system failures and hackers.

Massive - Database sizes in gigabytes, terabytes and petabytes.

Persistent - Data exists even if have a power failure.

Multi-user - More than one user can access and update data at the same time while preserving consistency.

Database Terminology

A ***data model*** is a collection of concepts that is used to describe the structure of a database. E.g. relational, XML, graphs, objects, JSON

- In the relational model, data is represented as tables and fields.

Data Definition Language (***DDL***) allows the user to create data structures in the data model used by the database. A ***schema*** is a description of the structure of the database and is maintained and stored in the ***system catalog***. The schema is ***metadata***.

- A schema contains structures, names, and types of data stored.

Once a database has been created using DDL, the user accesses data using a ***Data Manipulation Language*** (***DML***).

- The DML allows for the insertion, modification, retrieval, and deletion of data.

SQL is a standard DDL and DML for the relational model.

The Relational Model: Terminology

The ***relational model*** organizes data into tables called relations.

- Developed by E. F. Codd in 1970 and used by most database systems.

Terminology:

A ***relation*** is a table with columns and rows.

An ***attribute*** is a named column of a relation.

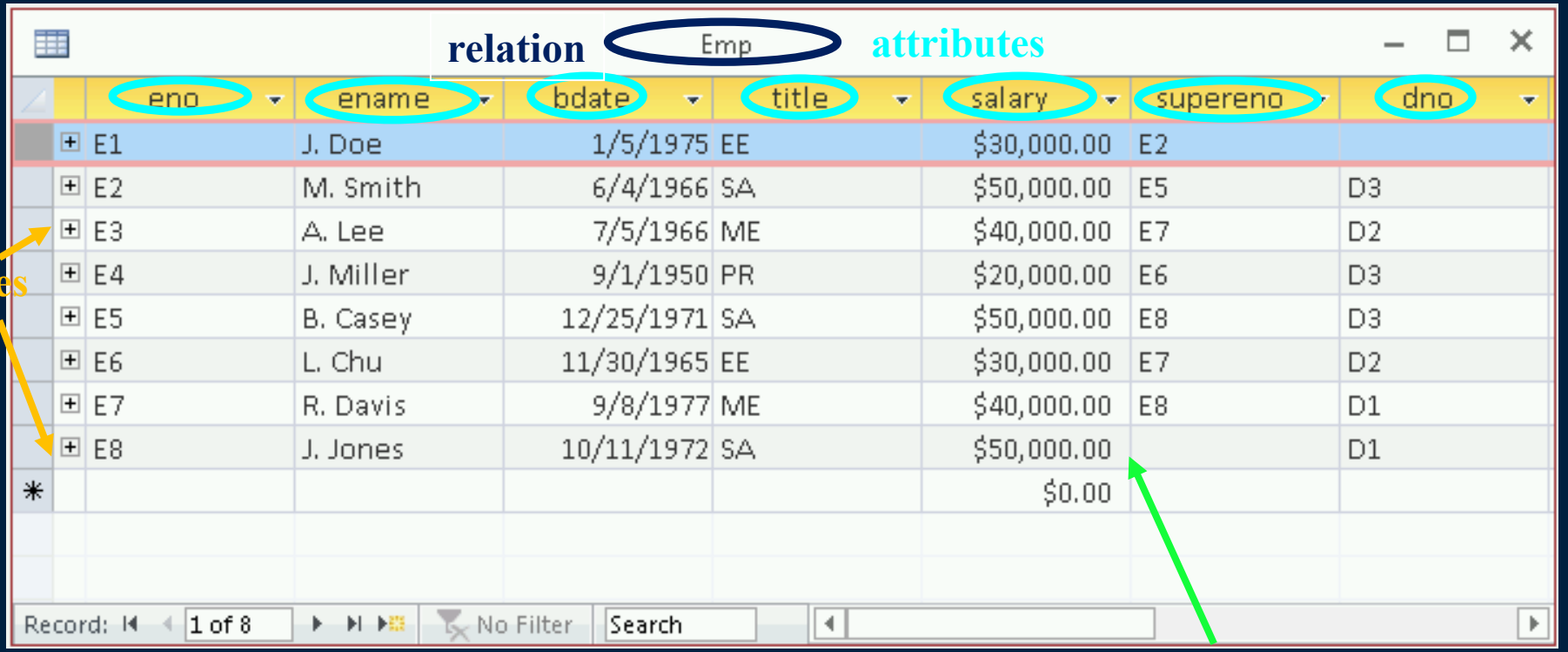
A ***tuple*** is a row of a relation.

A ***domain*** is a set of allowable values for one or more attributes.

The ***degree*** of a relation is the number of attributes it contains.

The ***cardinality*** of a relation is the number of tuples it contains.

Relation Example



eno	ename	bdate	title	salary	supereno	dno
E1	J. Doe	1/5/1975	EE	$30,000.00	E2	
E2	M. Smith	6/4/1966	SA	$50,000.00	E5	D3
E3	A. Lee	7/5/1966	ME	$40,000.00	E7	D2
E4	J. Miller	9/1/1950	PR	$20,000.00	E6	D3
E5	B. Casey	12/25/1971	SA	$50,000.00	E8	D3
E6	L. Chu	11/30/1965	EE	$30,000.00	E7	D2
E7	R. Davis	9/8/1977	ME	$40,000.00	E8	D1
E8	J. Jones	10/11/1972	SA	$50,000.00		D1
*				$0.00		

Degree =7
Cardinality = 8

Domain of salary is *currency*

Relation Practice Questions

WorksOn

eno	pno	resp	hours
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P3	Consultant	10
E3	P4	Engineer	48
E4	P2	Programmer	18
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36

Record: 1 of 9 No Filter Search

1) What is the name of the relation?
2) What is the cardinality of the relation?
3) What is the degree of the relation?
4) What is the domain of `resp`? What is the domain of `hours`?

Database Definition Question

Question: How many of the following statements are **TRUE**?

1) A database is data.
2) A database system is software.
3) A database system will lose the data stored when the power is turned off.
4) Usually, more than one user can use a database system at a time.
5) The cardinality is the number of rows in a relation.
6) A relation's cardinality is always bigger than its degree.

A) 1 **B)** 2 **C)** 3 **D)** 4 **E)** 5

Database Definition Matching Question

Question: Given the three definitions, select the ordering that contains their related definitions.

1) relation

2) tuple

3) attribute

A) column, row, table

B) row, column, table

C) table, row, column

D) table, column, row

Cardinality and Degree Question

Question: A database table has 5 rows and 10 columns. Select ***one*** true statement.

A) The table's degree is 50.

B) The table's cardinality is 5.

C) The table's degree is 5.

D) The table's cardinality is 10.

Relation Properties

1) No two relations have the same name.
2) Each attribute of a relation has a distinct name.
3) Each tuple is distinct. There are no duplicate tuples.
4) The order of attributes is not important.
5) The order of tuples has no significance.

Relational Keys

Keys are used to uniquely identify a tuple in a relation.

- Note that keys apply to the schema not to the data. That is, looking at the current data cannot tell you for sure if the set of attributes is a key.

A ***superkey*** is a set of attributes that uniquely identifies a tuple in a relation.

A (***candidate***) ***key*** (or minimal superkey) is a *minimal* set of attributes that uniquely identifies a tuple in a relation.

- There may be more than 1 candidate key for a relation with different # of attributes.

A ***primary key*** is the candidate key designated as the distinguishing key of a relation.

A ***foreign key*** is a set of attributes in one relation referring to the primary key of a relation.

- Foreign keys enforce referential integrity. Note: A FK may refer to its own relation.

Keys and Superkeys Question

Question: **True or false:** A key is always a superkey.

A) true

B) false

Keys and Superkeys Question (2)

Question: **True or false:** It is possible to have more than one key for a table and the keys may have different numbers of attributes.

A) true

B) false

Keys and Superkeys Question (3)

Question: **True or false:** It is possible to always determine if a field is a key by looking at the data in the table.

A) true

B) false

Example Relational Data Questions

Emp Relation

eno	ename	title	salary
E1	J. Doe	EE	30000
E2	M. Smith	SA	50000
E3	A. Lee	ME	40000
E4	J. Miller	PR	20000
E5	B. Casey	SA	50000
E6	L. Chu	EE	30000
E7	R. Davis	ME	40000
E8	J. Jones	SA	50000

WorksOn Relation

eno	pno	resp	dur
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P3	Consultant	10
E3	P4	Engineer	48
E4	P2	Programmer	18
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36
E7	P5	Engineer	23
E8	P3	Manager	40

Proj Relation

pno	pname	budget
P1	Instruments	150000
P2	DB Develop	135000
P3	CAD/CAM	250000
P4	Maintenance	310000
P5	CAD/CAM	500000

Questions:

1) Is *ename* a key for *emp*?
2) Is *eno* a key for *WorksOn*?
3) List all the superkeys for *WorksOn*.

Relational Integrity

Integrity rules are used to insure the data is accurate.

Constraints are rules or restrictions that apply to the database and limit the data values it may store.

Types of constraints:

- ***Domain constraint*** - Every value for an attribute must be an element of the attribute's domain or be `null`.
  - `null` represents a value that is currently unknown or not applicable.
  - `null` is not the same as zero or an empty string.
- ***Entity integrity constraint*** - No attribute of a primary key can be null.
- ***Referential integrity constraint*** - If a foreign key exists in a relation, then the foreign key value must match a primary key value of a tuple in the referenced relation or be null.

Foreign Keys Example

Emp Relation

eno	ename	title	salary
E1	J. Doe	EE	30000
E2	M. Smith	SA	50000
E3	A. Lee	ME	40000
E4	J. Miller	PR	20000
E5	B. Casey	SA	50000
E6	L. Chu	EE	30000
E7	R. Davis	ME	40000
E8	J. Jones	SA	50000

WorksOn Relation

eno	pno	resp	dur
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P3	Consultant	10
E3	P4	Engineer	48
E4	P2	Programmer	18
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36
E7	P5	Engineer	23
E8	P3	Manager	40

WorksOn.eno is FK to Emp.eno

WorksOn.pno is FK to Proj.pno

Proj Relation

pno	pname	budget
P1	Instruments	150000
P2	DB Develop	135000
P3	CAD/CAM	250000
P4	Maintenance	310000
P5	CAD/CAM	500000

Foreign Keys Example (2)

Proj Relation

pno	pname	budget	dno
P1	Instruments	150000	D1
P2	DB Develop	135000	D2
P3	CAD/CAM	250000	D3
P4	Maintenance	310000	null
P5	CAD/CAM	500000	D1

We can later assign this project to any of the existing departments in the Department Relation. But can't assign it to, say, D6 because it doesn't exist in the Department table.

Proj.dno is FK to Dept.dno

Department Relation

dno	dname
D1	Management
D2	Consulting
D3	Accounting
D4	Development

Integrity Constraints Question

Question: What constraint says that a primary key field cannot be null?

A) domain constraint

B) referential integrity constraint

C) entity integrity constraint

Entity Integrity Constraint Question

Question: A primary key has three fields. Only one field is `null`. Is the entity integrity constraint violated?

A) Yes

B) No

Referential Integrity Constraint Question

Question: A foreign key has a `null` value in the table that contains the foreign key fields. Is the referential integrity constraint violated?

A) Yes

B) No

Integrity Questions

Emp Relation

eno	ename	title	salary
E1	J. Doe	EE	AS
E2	null	SA	50000
E3	A. Lee	null	40000
E4	J. Miller	PR	20000
E5	B. Casey	SA	50000
null	L. Chu	EE	30000
E7	R. Davis	ME	null
E8	J. Jones	SA	50000

Proj Relation

pno	pname	budget
P1	Instruments	150000
P2	DB Develop	135000
P3	CAD/CAM	250000
P4	Maintenance	310000
P5	null	null

WorksOn Relation

eno	pno	resp	dur
E1	P0	null	12
E2	P1	Analyst	null
null	P2	Analyst	6
E3	P3	Consultant	10
E9	P4	Engineer	48
E4	P2	Programmer	18
E5	null	Manager	24
E6	P4	Manager	48
E7	P6	Engineer	36
E7	P4	Engineer	23
null	null	Manager	40

Question: How many rows have violations of integrity constraints? Note: salary, budget, dur are number fields.

A) 8 B) 9 C) 10 D) 11 E) 12

Relational Algebra Query Language

A ***query language*** is used to update and retrieve data that is stored in a data model.

Relational algebra is a set of relational operations for retrieving data.

- Just like algebra with numbers, relational algebra consists of operands (which are relations) and a set of operators.

Every relational operator takes as input one or more relations and produces a relation as output.

$\sigma_p(r)$: σ subject = "database"(Books)

A sequence of relational algebra operators is called a ***relational algebra expression***.

Relational algebra is the foundation of all relational database systems. SQL gets translated into relational algebra.

Conclusion

A ***database*** is a collection of logically related data managed by a ***database management system*** (DBMS).

- Provides data independence and abstraction
- Data definition and manipulation languages (DDL and DML)

The ***relational model*** represents data as relations which are sets of tuples. Each relational schema is a set of attributes with domains.

The relational model has ***constraints*** to guarantee data integrity including: domain, entity integrity and referential integrity constraints. ***Keys*** are used to uniquely identify tuples in relations.

Relational algebra is a set of operations for answering queries on data stored in the relational model.


UBC
THE UNIVERSITY OF BRITISH COLUMBIA