DATA 301
Introduction to Data Analytics
Relational Databases

Dr. Ramon Lawrence
University of British Columbia Okanagan
ramon.lawrence@ubc.ca

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?

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.

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.

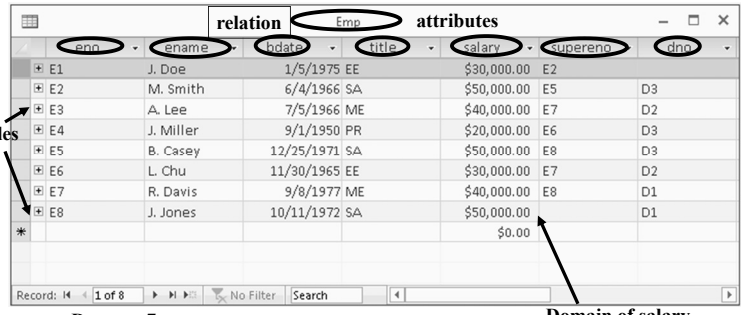
A ***tuple*** is a row of a relation.

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.

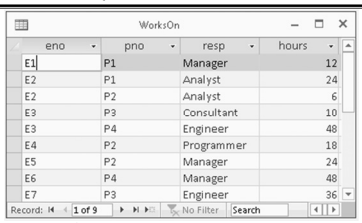
Relation Example



Degree =7
Cardinality = 8

Domain of salary is *currency*

Relation Practice Questions



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) 0 **B)** 1 **C)** 2 **D)** 3 **E)** 4

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.

Creating and Using a Database

Typically, a data analyst will use an existing database. The database will already be created on a database system and contain data that was inserted and updated previously.

To use an existing database, the data analyst must be able to use the tools and languages to query the database. The standard is SQL.

Creating a large database is outside of the scope of this class, but we will learn how to create individual tables and load data into them which is a common data analysis task.

A Simple Query Language: Keyword Searching

Keyword (or English-language) ***search*** allows a user to type keywords or phrases and returns a best answer estimate.

This works fairly well for web searches, although we lack precision. Precision is required for many applications.

- Example: How would you return all employees with salary greater than 30,000 using keyword search?

SQL Overview

Structured Query Language or SQL is the standard database query language to retrieve *exact answers*.

- A SQL query specifies *what* to retrieve but not *how* to retrieve it.
- SQL is used by Microsoft Access and almost all other database systems.

Some basic rules for SQL statements:

- 1) There is a set of *reserved words* that cannot be used as names for database fields and tables.
  - `SELECT`, `FROM`, `WHERE`, etc.
- 2) SQL is generally *case-insensitive*.
  - Only exception is string constants. 'FRED' not the same as 'fred'.
- 3) SQL is *free-format* and white-space is ignored.

SQL `CREATE TABLE`

The **`CREATE TABLE`** command is used to create a table in the database. A table consists of a table name and a set of fields with their names and data types.

Example:

```
CREATE TABLE emp (
    eno       CHAR(5),
    ename     VARCHAR(30) NOT NULL,
    bdate     DATE,
    title     CHAR(2),
    salary    DECIMAL(9,2),
    supereno  CHAR(5),
    dno       CHAR(5),
    PRIMARY KEY (eno)
)
```

field must always have a value

Data Types:
CHAR(5) – always 5 chars long
VARCHAR(30) – up to 30 chars long
DECIMAL(9,2) – e.g. 1234567.99
DATE – e.g. 1998/01/18

What is a key?

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

A key helps to identify a particular row (data item) and find it faster.

In the `emp` table, the key was `eno`. It was called the `primary key` because it was the main key used to find an employee in the table.

Question:

- What is a key to identify a student in this class?

Try it: `CREATE TABLE`

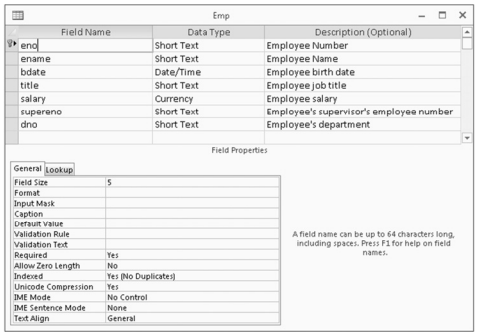
Question: Create a table called `mydata` that has three fields:

- `num` – that will store a number (use `int` as data type)
- `message` – that will store a string up to 50 characters (`varchar` data type)
- `amount` – that stores a decimal number with 8 total digits and 2 decimal digits (`decimal` data type)

Use the web site **sqlfiddle.com** to try your table creation.

`CREATE TABLE` in Microsoft Access

In Microsoft Access, use `Create` -> `Table` to build a table.



Try it: `CREATE TABLE` in Microsoft Access

Question: Create a table called `mydata` that has three fields:

- `num` – that will store a number (use `Number` as data type)
- `message` – that will store a string up to 50 characters (`Short Text` data type)
- `amount` – that stores a decimal number with 8 total digits and 2 decimal digits (`Currency` data type)

Build the table using the Microsoft Access user interface.

Schemas and Metadata

Creating tables defines the structure of the database.

The description of the structure of the database is called a ***schema***.

The schema is a type of ***metadata***.

`DROP TABLE`

The command **`DROP TABLE`** is used to delete the table and *all its data* from the database:

Example:

```
DROP TABLE emp;
```

- Note: The database does not confirm if you really want to drop the table and delete its data. The effect of the command is immediate.

`CREATE TABLE` Question

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

1) Each field in the `CREATE TABLE` statement is separated by a comma.
2) The data type for a field is optional.
3) You can create two tables in a database with the same name.
4) A table will not be dropped (with `DROP TABLE`) if it contains data.

A) 0 **B)** 1 **C)** 2 **D)** 3 **E)** 4

Adding Data using `INSERT`

Insert a row using the `INSERT` command:

```
INSERT INTO emp VALUES ('E9','S. Smith','1975-03-05',
                        'SA',60000,'E8','D1')
```

Fields: eno, ename, bdate, title, salary, supereno, dno

If you do not give values for all fields in the order they are in the table, you must list the fields you are providing data for:

```
INSERT INTO emp(eno, ename, salary)
       VALUES ('E9','S. Smith',60000)
```

Note: If any columns are omitted from the list, they are set to `NULL` (empty).

Try it: `INSERT`

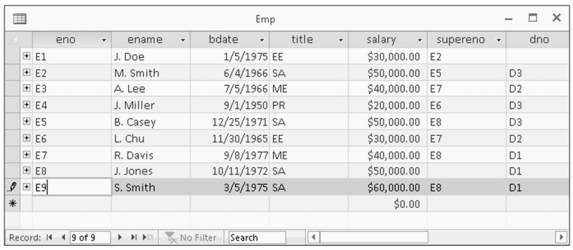
Question: Using the `mydata` table insert three rows:

- `(1, 'Hello', 99.45)`
- `(2, 'Goodbye', 55.99)`
- `(3, 'No Amount')`

Use the web site **sqlfiddle.com** to try your table creation.

Adding Data using `INSERT` in Microsoft Access

In Microsoft Access, insert a new row by entering data into the last row of the table when in data view.



Try it: `INSERT` in Microsoft Access

Question: Using the `mydata` table insert three rows in Access:

- `(1, 'Hello', 99.45)`
- `(2, 'Goodbye', 55.99)`
- `(3, 'No Amount')`

`UPDATE` Statement

Updating existing rows using the `UPDATE` statement. Examples:

- 1) Increase all employee salaries by 10%.

```
UPDATE emp SET salary = salary*1.10;
```

- 2) Increase salary of employee E2 to $1 million and change his name:

```
UPDATE emp SET salary = 1000000, name='Rich Guy'
WHERE eno = 'E2';
```

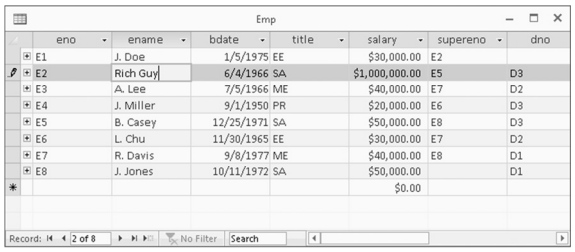
Notes:

- May change (`SET`) more than one value at a time. Separate by commas.
- Use `WHERE` to filter only the rows to update.

Updating Data in Microsoft Access

`UPDATE` command supported by Microsoft Access.

To modify individual data items, select the row and cell to update and change the data. Data is saved when you leave the row.



Try it: `UPDATE`

Question: Using the `mydata` table and the three rows previously inserted do these updates:

- Update all `amount` fields to be `99.99`.
- Update the `num` field and set it to 10 for the record with `num = 1`.
- Update the `message` field to `'Changed'` for the record with `num = 2`.

You can use Access or sqlfiddle.com.

`DELETE` Statement

Rows are deleted using the `DELETE` statement. Examples:

- 1) Fire everyone in the company.

```
DELETE FROM emp;
```

- 2) Fire everyone making over $35,000.

```
DELETE FROM emp
WHERE salary > 35000;
```

Deleting Data in Microsoft Access

`DELETE` command supported by Microsoft Access. To delete an individual row, select the row to delete and press `Delete` key or select `Delete Record` from pop-up menu.

Try it: `DELETE`

Question: Using the `mydata` table and the three rows previously inserted do these deletes:

- Delete the row with `num = 1`.
- Delete the row(s) with `message > 'C'`.
- Delete all rows.

You can use Access or sqlfiddle.com.

`INSERT` Question

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

1) You must always specify the fields being inserted with `INSERT` statement.
2) If you list the fields, the fields must be in the same order as the table.
3) If you do not provide a value for a number field, it will default to 1.
4) Number data items are enclosed in single quotes.

A) 0 **B)** 1 **C)** 2 **D)** 3 **E)** 4

`UPDATE` Question

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

1) You may update more than one row at a time.
2) If the `UPDATE` has no `WHERE` clause, it always updates all rows.
3) You may update zero or more rows using a `UPDATE` statement.
4) `UPDATE` may change more than one data value (column) in a row.

A) 0 **B)** 1 **C)** 2 **D)** 3 **E)** 4

`DELETE` Question

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

1) A `DELETE` with no `WHERE` clause will delete all rows.
2) `DELETE` statement is case-sensitive.
3) It is possible to `DELETE` zero or more rows using a `WHERE` clause.
4) A `DELETE` statement may delete zero rows when executed.

A) 0 **B)** 1 **C)** 2 **D)** 3 **E)** 4

SQL Queries using `SELECT`

A query in SQL has the form:

SELECT (list of columns or expressions)
FROM (list of tables)
WHERE (filter *conditions*)
GROUP BY (columns)
ORDER BY (columns)

Notes:

- 1) Separate the list of columns/expressions and list of tables by **commas**.
- 2) The "`*`" is used to select all columns.
- 3) Only `SELECT` required. `FROM`, `WHERE`, `GROUP BY`, `ORDER BY` are optional.

Example Data

emp Table

eno	ename	bdate	title	salary	supereno	dno
E1	J. Doe	01-05-75	EE	30000	E2	null
E2	M. Smith	06-04-66	SA	50000	E5	D3
E3	A. Lee	07-05-66	ME	40000	E7	D2
E4	J. Miller	09-01-50	PR	20000	E6	D3
E5	B. Casey	12-25-71	SA	50000	E8	D3
E6	L. Chu	11-30-65	EE	30000	E7	D2
E7	R. Davis	09-08-77	ME	40000	E8	D1
E8	J. Jones	10-11-72	SA	50000	null	D1

proj Table

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

workson Table

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

dept Table

dno	dname	mgreno
D1	Management	E8
D2	Consulting	E7
D3	Accounting	E5
D4	Development	null

SQL: Retrieving Only Some of the Columns

The ***projection operation*** creates a new table that has some of the columns of the input table. In SQL, provide the table in the `FROM` clause and the fields in the output in the `SELECT`.

Example: Return only the `eno` field from the `Emp` table:

```
SELECT eno
FROM   emp
```

emp Table

eno	ename	bdate	title	salary	supereno	dno
E1	J. Doe	01-05-75	EE	30000	E2	null
E2	M. Smith	06-04-66	SA	50000	E5	D3
E3	A. Lee	07-05-66	ME	40000	E7	D2
E4	J. Miller	09-01-50	PR	20000	E6	D3
E5	B. Casey	12-25-71	SA	50000	E8	D3
E6	L. Chu	11-30-65	EE	30000	E7	D2
E7	R. Davis	09-08-77	ME	40000	E8	D1
E8	J. Jones	10-11-72	SA	50000	null	D1

Result

eno
E1
E2
E3
E4
E5
E6
E7
E8

SQL Projection Examples

emp Table

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

```
SELECT eno,ename
FROM   emp
```

eno	ename
E1	J. Doe
E2	M. Smith
E3	A. Lee
E4	J. Miller
E5	B. Casey
E6	L. Chu
E7	R. Davis
E8	J. Jones

```
SELECT title
FROM   emp
```

title
EE
SA
ME
PR
SA
EE
ME
SA

Notes: 1) Duplicates are not removed during SQL projection.
2) `SELECT *` will return all columns.

Projection Question

Question: Given this table and the query:

```
SELECT eno, ename, salary
FROM   emp
```

How many columns are returned?

A) 0

B) 1

C) 2

D) 3

E) 4

emp Table

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

Projection Question #2

Question: Given this table and the query:

```
SELECT salary
FROM   emp
```

How many rows are returned?

A) 0

B) 2

C) 4

D) 8

emp Table

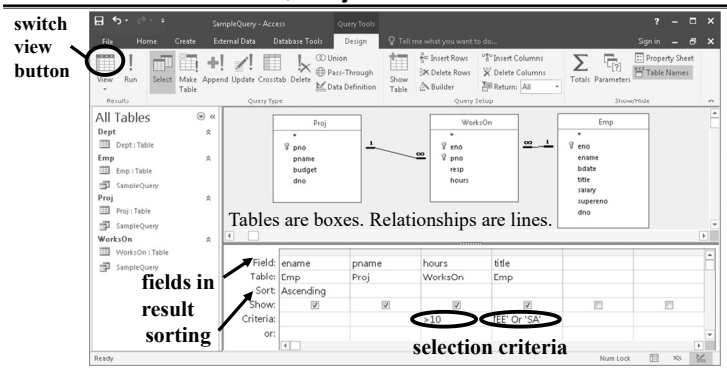
eno	ename	title	salary
E1	J. Doe	EE	30000
E2	M. Smith	SA	50000
E3	A. Lee	ME	40000
E4	J. Miller	PR	20000
E5	B. Casey	SA	50000
E6	L. Chu	EE	30000
E7	R. Davis	ME	40000
E8	J. Jones	SA	50000

Building a `SELECT` SQL Query in Microsoft Access

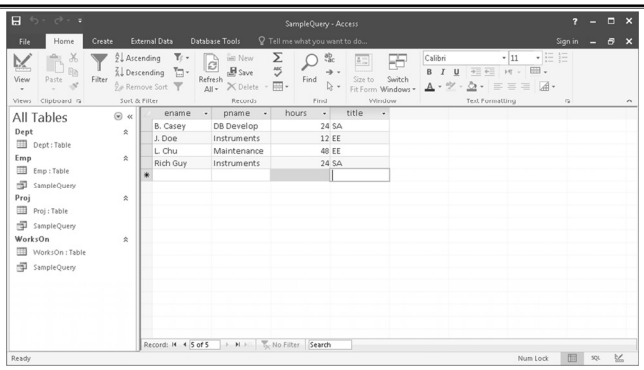
Under `Create Tab`, click on `Query Design`.

Access will pop-up a window asking what table(s) you wish to query. Select one or more.

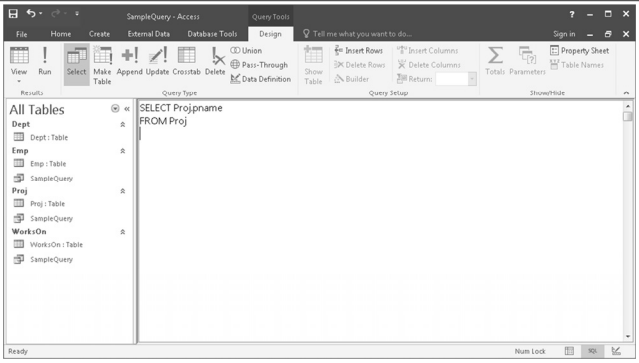
Microsoft Access Query Interface



Microsoft Access Data Sheet View

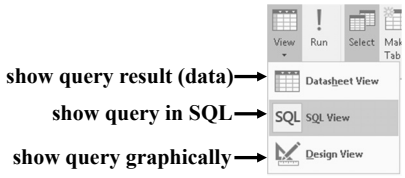


Microsoft Access SQL Design View



Microsoft Access Query Views

You may view your data, your query graphically, or your query in SQL.



Try it: SQL `SELECT` and Projection

Question: Using the `proj` table, write these three queries:

- Show all rows and all columns.
- Show all rows but only the `pno` column.
- Show all rows but only the `pno` and `budget` columns.

You can use Access or sqlfiddle.com.

Retrieving Only Some of the Rows

The ***selection operation*** creates a new table with some of the rows of the input table. A condition specifies which rows are in the new table. The condition is similar to an `if` statement.

Example: Return the projects in department `'D2'`:

```
SELECT pno, pname, budget, dno
FROM   proj
WHERE  dno = 'D2';
```

`proj` Table

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

Result

pno	pname	budget	dno
P2	DB Develop	135000	D2
P4	Maintenance	310000	D2
P5	CAD/CAM	500000	D2

Algorithm: Scan each tuple and check if matches condition in WHERE clause.

Selection Conditions

The condition in a selection statement specifies which rows are included. It has the general form of an if statement.

The condition may consist of attributes, constants, comparison operators (<, >, =, !=, <=, >=), and logical operators (AND, OR, NOT).

SQL Selection Examples

emp Table

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

```
SELECT *
FROM   emp
WHERE  title = 'EE'
```

eno	ename	title	salary
E1	J. Doe	EE	30000
E6	L. Chu	EE	30000

```
SELECT eno, ename, title, salary
FROM   emp
WHERE  salary > 35000 OR
       title = 'PR'
```

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

Selection Question

Question: Given this table and the query:

```
SELECT *
FROM   emp
WHERE  title='SA'
```

How many rows are returned?

A) 0

B) 1

C) 2

D) 3

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

Selection Question #2

Question: Given this table and the query:

```
SELECT *
FROM   emp
WHERE  salary > 50000 or title='PR'
```

How many rows are returned?

A) 0

B) 1

C) 2

D) 3

emp Table

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

Selection Question #3

Question: Given this table and the query:

```
SELECT *
FROM   emp
WHERE  salary > 50000 or title='PR'
```

How many columns are returned?

A) 0 **B)** 1 **C)** 2 **D)** 3 **E)** 4

emp Table

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

Try it: SQL SELECT and Filtering Rows

Question: Using the proj table, write these three queries:

- Return all projects with budget > $250000.
- Show the pno and pname for projects in dno = 'D1'.
- Show pno and dno for projects in dno='D1' or dno='D2'.

You can use Access or sqlfiddle.com.

Join Example for Combining Tables

A ***join*** combines two tables by matching columns in each table.

workson Table

eno	pno	resp	dur
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P4	Engineer	48
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36
E7	P4	Engineer	23

proj Table

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

```
SELECT *
FROM   WorksOn INNER JOIN Proj
  ON   WorksOn.pno = Proj.pno
```

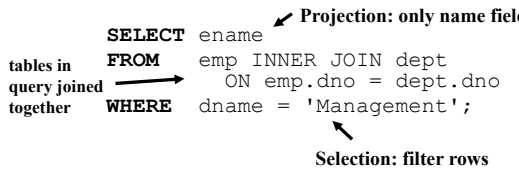
eno	pno	resp	dur	P.pno	pname	budget
E1	P1	Manager	12	P1	Instruments	150000
E2	P1	Analyst	24	P1	Instruments	150000
E2	P2	Analyst	6	P2	DB Develop	135000
E3	P4	Engineer	48	P4	Maintenance	310000
E5	P2	Manager	24	P2	DB Develop	135000
E6	P4	Manager	48	P4	Maintenance	310000
E7	P3	Engineer	36	P3	CAD/CAM	250000
E7	P4	Engineer	23	P4	Maintenance	310000

Join Query with Selection Example

You can use join, selection, and projection in the same query.

- Recall: Projection returns columns listed in `SELECT`, selection filters out rows using condition in `WHERE`, and join combines tables in `FROM` using a condition.

Example: Return the employee names who are assigned to the 'Management' department.



Result

ename
R. Davis
J. Jones

Ordering Result Data

The query result returned is not ordered on any column by default. We can order the data using the **`ORDER BY`** clause:

```
SELECT   ename, salary, bdate
FROM     emp
WHERE    salary > 30000
ORDER BY salary DESC, ename ASC;
```

- '`ASC`' sorts the data in ascending order, and '`DESC`' sorts it in descending order. The default is '`ASC`'.
- The order of sorted attributes is significant. The first column specified is sorted on first, then the second column is used to break any ties, etc.

`LIMIT` and `OFFSET`

If you only want the first *N* rows, use a **`LIMIT`** clause:

```
SELECT   ename, salary FROM emp
ORDER BY salary DESC LIMIT 5
```

To start from a row besides the first, use **`OFFSET`**:

```
SELECT   eno, salary FROM emp
ORDER BY eno DESC
LIMIT 3  OFFSET 2
```

- `LIMIT` improves performance by reducing amount of data processed and sent by the database system.
- `OFFSET` 0 is first row, so `OFFSET` 2 would return the 3rd row.
- `LIMIT`/`OFFSET` syntax supported differently by systems.
- For Access, use `SELECT TOP 5 eno, salary FROM emp`

Try it: SQL `SELECT` with Joins and Ordering

Question: Write these three queries:

- Return all projects with `budget` < $`500000` sorted by `budget` descending.
- List only the top 5 employees by `salary` descending. Show only their `name` and `salary`.
- List each project `pno`, `dno`, `pname`, and `dname` ordered by `dno` ascending then `pno` ascending. Only show projects if department `name > 'D'`. Note: This query will require a join.

You can use Access or sqlfiddle.com.

Aggregate Queries and Functions

Several queries cannot be answered using the simple form of the `SELECT` statement. These queries require a summary calculation to be performed. Examples:

- What is the maximum employee salary?
- What is the total number of hours worked on a project?
- How many employees are there in department 'D1'?

To answer these queries requires the use of aggregate functions. These functions operate on a single column of a table and return a single value.

Aggregate Functions

Five common aggregate functions are:

- `COUNT` - returns the # of values in a column
- `SUM` - returns the sum of the values in a column
- `AVG` - returns the average of the values in a column
- `MIN` - returns the smallest value in a column
- `MAX` - returns the largest value in a column

Notes:

- 1) `COUNT`, `MAX`, and `MIN` apply to all types of fields, whereas `SUM` and `AVG` apply to only numeric fields.
- 2) Except for `COUNT(*)` all functions ignore nulls. `COUNT(*)` returns the number of rows in the table.
- 3) Use `DISTINCT` to eliminate duplicates.

Aggregate Function Example

Return the number of employees and their average salary.

```
SELECT COUNT(eno) AS numEmp, AVG(salary) AS avgSalary
FROM   emp
```

Result

numEmp	avgSalary
8	38750

Note: `AS` is used to rename a column in the output.

GROUP BY Clause

Aggregate functions are most useful when combined with the `GROUP BY` clause. The **`GROUP BY`** clause groups rows based on the values of the columns specified.

When used in combination with aggregate functions, the result is a table where each row consists of unique values for the group by attributes and the result of the aggregate functions applied to the rows of that group.

GROUP BY Example

For each employee title, return the number of employees with that title, and the minimum, maximum, and average salary.

```
SELECT   title, COUNT(eno) AS numEmp,
         MIN(salary) as minSal,
         MAX(salary) as maxSal, AVG(salary) AS avgSal
FROM     emp
GROUP BY title
```

Result

title	numEmp	minSal	maxSal	avgSal
EE	2	30000	30000	30000
SA	3	50000	50000	50000
ME	2	40000	40000	40000
PR	1	20000	20000	20000

GROUP BY Facts

1) You can group by multiple attributes. To be in the same group, all attribute values must be the same.

2) Any `WHERE` conditions are applied before the `GROUP BY` and aggregate functions are calculated.

3) A column name cannot appear in the `SELECT` part of the query unless it is part of an aggregate function or in the list of group by attributes.

4) There is a `HAVING` clause that is applied *AFTER* the `GROUP BY` clause and aggregate functions are calculated to filter out groups. (We will not study that.)

GROUP BY Question

Question: Given this table and the query:

```
SELECT title, SUM(salary)
FROM   emp
GROUP BY title
```

How many rows are returned?

A) 1

B) 2

C) 4

D) 8

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

GROUP BY Question #2

Question: Given this table and the query:

```
SELECT resp, pno, SUM(hours)
FROM   workson
WHERE  hours > 10
GROUP BY resp, pno
```

workson Table

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

How many rows are returned?

A) 9 **B)** 7 **C)** 5 **D)** 1 **E)** 0

Try it: GROUP BY

Question: Use GROUP BY and aggregation functions to answer these queries.

1) Output the number of projects in the database.

2) Return the sum of the budgets for all projects.

3) For each department (dno), return the department number (dno) and the average budget of projects in that department.

4) For each project (pno), return the project number (pno) and the sum of the number of hours employees have worked on that project.

- Challenge: Show the project name (pname) as well as the project number.

5) Challenge: Show the department name (dname), project name (pname), and sum of hours worked on that project as well as the number of employees working on the project.

You can use Access or sqlfiddle.com.

Putting it All Together

The steps to write an English query in SQL are:

- 1) Find the columns that you need and put in SELECT clause.
- 2) List the tables that have the columns in the FROM clause. If there is more than one, join them together.
- 3) If you must filter rows, add a filter criteria in WHERE clause.
- 4) If you need to create an aggregate, use aggregation functions and GROUP BY.

Example: For each project name list the sum of the hours worked by employees working as a 'Manager' on the project.

```
SELECT pname, SUM(hours) as totalHours
FROM   workson INNER JOIN proj on workson.pno=proj.pno
WHERE  resp='Manager'
GROUP BY pname
```


Microsoft Access Querying Summary

1) Projection is performed by selecting the fields in the output in the field row in the table at the bottom of the screen.

2) Selection is performed by entering the condition in the criteria box. The criteria applies to the field in that column.

3) The tables used are added to the query by the **Show Table**… option.

4) Joins (based on relationships) are often automatically added, but if not, you can add them by selecting the join field in one table, holding the mouse button, then dragging to the join field in the other table.

Conclusion

A ***database*** is a collection of related data. A ***database system*** allows storing and querying a database.

SQL is the standard query language for databases, although Microsoft Access also provides a graphical user interface.

CREATE TABLE creates a table. INSERT, DELETE, and UPDATE commands modify the data stored within the database.

The basic query operations are selection (subset of rows), projection (subset of columns), join (combine two or more tables), and grouping and aggregation.

Objectives

- Define: database, database system, schema, metadata
- Define: relation, attribute, tuple, domain, degree, cardinality
- SQL properties: reserved words, case-insensitive, free-format
- Be able to create a table using CREATE TABLE command and in Microsoft Access.
- Explain what a key is and what it is used for.
- Use DROP TABLE to delete a table and its data.
- Use INSERT/UPDATE/DELETE to add/update/delete rows of a table and perform same actions using Microsoft Access user interface.
- Execute queries using SQL SELECT and using Microsoft Access user interface.
- Sort rows using ORDER BY. Use LIMIT to keep only the first (top) N rows.
- Use GROUP BY and aggregation functions for calculating summary data.

☆Given a small database write simple English queries in SQL.