>>> Introduction to SQL
>>> Featuring MySQL and T-SQL

Daniel Fryer [†]

Feb, 2023

[†] daniel@vfryer.com

>>> Daily schedule

Timetable				
9:00am	-	10:30am	lecture 1	(1.5 hr)
10:30am	-	11:00am	morning tea	(30 min)
11:00am	-	12:30pm	lecture 2	(1.5 hr)
12:30pm	-	1:30pm	lunch	(1 hr)
1:30pm	-	3:00pm	breakout / windowing	(1.5 hr)
3:00pm	-	5:00pm	one-on-one help	(2 hr)

>>> Where are we now?

Day 3

1. Recap of days 1 and 2
2. Expanding the toolkit
3. Creating data
4. Working with CSVs
5. The IDI
6. Independent development
7. Windowing part 2

   Send me questions and give feedback

Page is hyperlinked: click a topic above to jump to it.

>>> Kahoot! Recap of days 1 and 2

Enjoy.

>>> Where are we now?

Day 3

1. Recap of days 1 and 2
2. Expanding the toolkit
3. Creating data
4. Working with CSVs
5. The IDI
6. Independent development
7. Windowing part 2

   Send me questions and give feedback

Page is hyperlinked: click a topic above to jump to it.

>>> Group practice

Figure out what UNION does.

- * Click here for T-SQL UNION documentation.
- * Click here for MySQL UNION documentation.

>>> Live demonstration of UNION

Starting with this...

```
SELECT F.FirstName AS FirstInitial,
       F.LastName AS LastInitial,
       ColourName
FROM Ape.Friends F LEFT JOIN Ape.Colours C
     ON F.FavColourID = C.ColourID;
```

>>> Group practice

Figure out what FORMAT (T-SQL) and DATE_FORMAT (MySQL) do.

* Click here for T-SQL FORMAT docs.
* Click here for MySQL DATE_FORMAT docs.

>>> Live demonstration of date formatting

Let's format some dates.

>>> Group practice

Figure out what COUNT and COUNT(DISTINCT) do.

- Click here for T-SQL COUNT documentation.
- Click here for MySQL COUNT documentation.

>>> Live demo

Starting with this...

```
SELECT COUNT(TasteRank)
FROM Ape.Banana;
```

>>> Group practice

Figure out what WITH does.

- Click here for T-SQL WITH documentation.
- Click here for MySQL WITH documentation.

>>> Live demo

Reducing repetition via WITH.

```
SELECT BananaID, TasteRank, Ripe,
      CASE WHEN Ripe = 1 AND TasteRank = 5 THEN 'Ripe and tasty'
      ELSE 'Imperfect' END AS Category
FROM Ape.Banana
GROUP BY Category;
```

>>> Many other functions

There are many other functions...

* Click here for T-SQL functions
* Click here for MySQL functions

Check out the notes section on window functions.

>>> Where are we now?

Day 3

1. Recap of days 1 and 2
2. Expanding the toolkit
3. Creating data
4. Working with CSVs
5. The IDI
6. Independent development
7. Windowing part 2

   Send me questions and give feedback

Page is hyperlinked: click a topic above to jump to it.

>>> Live demonstration

* CREATE DATABASE and DROP DATABASE
* CREATE SCHEMA (for T-SQL only)
* CREATE VIEW to store a query like a table
* SELECT INTO and CREATE TABLE ... SELECT
* INSERT INTO to create a whole record
* CREATE TABLE and DROP TABLE
* ALTER to add columns to a stored table
* UPDATE to change the entries in a table

You can also see all of the above in the notes.

>>> Where are we now?

Day 3

1. Recap of days 1 and 2
2. Expanding the toolkit
3. Creating data
4. Working with CSVs
5. The IDI
6. Independent development
7. Windowing part 2

   Send me questions and give feedback

Page is hyperlinked: click a topic above to jump to it.

>>> Live demonstration

Importing/exporting CSVs in various editors.

>>> Where are we now?

Day 3

1. Recap of days 1 and 2
2. Expanding the toolkit
3. Creating data
4. Working with CSVs
5. The IDI
6. Independent development
7. Windowing part 2

Send me questions and give feedback

Page is hyperlinked: click a topic above to jump to it.

>>> How we cover the IDI

- This is not a full intro to the IDI
- Our queries probably won't work on the IDI
- I'll make some important points (from a SQL perspective)
- The IDI uses T-SQL

>>> What is the IDI?

A collection of databases and schemas containing deidentified administrative and survey data from people's interactions with many government departments.

The different government departments use different unique identifiers, so interactions have been linked to individuals probabilistically.

>>> What is the IDI?

A collection of databases and schemas containing deidentified administrative and survey data from people's interactions with many government departments.

The different government departments use different unique identifiers, so interactions have been linked to individuals probabilistically.

The schemas (sometimes called nodes) in the main database correspond mostly to different government departments. From a technical perspective, the probabilistic linking allows us to JOIN records between schemas.

>>> Databases in the IDI

* IDI_Clean (and previous 'refreshes')
* IDI_Metadata
* IDI_Sandpit
* IDI_RnD

>>> IDI SQL Server

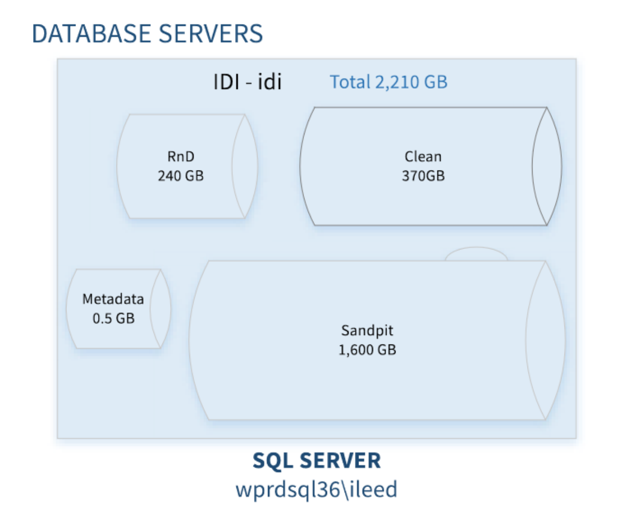
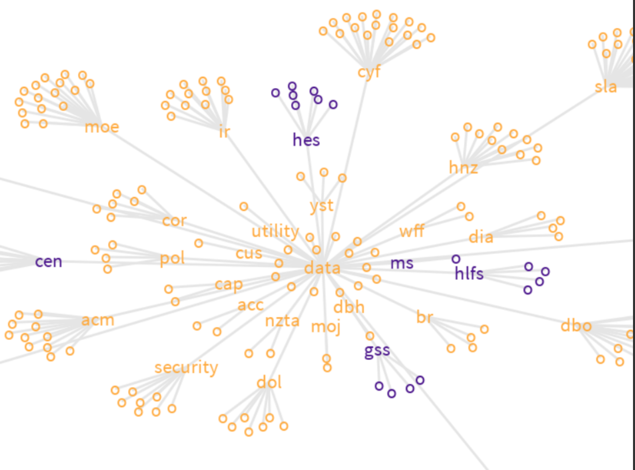


Image source: Use of IDI, Peter Ellis.

>>> Zooming in to IDI_Clean

Survey (purple) and admin (orange) data.



Schemas are mainly for permission managament.

>>> Zooming back out: the spine

The spine is a derived dataset that we don't have access to. It is deemed by Stats NZ to be the most ideal for identifying an ever-resident population.

>>> Zooming back out: the spine

The spine is a derived dataset that we don't have access to. It is deemed by Stats NZ to be the most ideal for identifying an ever-resident population.

Individuals in the spine are the only ones whose records are linked. All links are made "through the spine." There are 10 mil people in the spine, and 57 mil people in the IDI.

>>> Zooming back out: the spine

The spine is a derived dataset that we don't have access to. It is deemed by Stats NZ to be the most ideal for identifying an ever-resident population.

Individuals in the spine are the only ones whose records are linked. All links are made "through the spine." There are 10 mil people in the spine, and 57 mil people in the IDI.

A good spine should include every person in the target population once and only once. It includes tax, births and visa data (not deidentified).

>>> Zooming back out: the spine

The spine is a derived dataset that we don't have access to. It is deemed by Stats NZ to be the most ideal for identifying an ever-resident population.

Individuals in the spine are the only ones whose records are linked. All links are made "through the spine." There are 10 mil people in the spine, and 57 mil people in the IDI.

A good spine should include every person in the target population once and only once. It includes tax, births and visa data (not deidentified).

- Permanent residents
- Visas to reside, work or study
- People that live and work here without visas (e.g., Australians)

>>> More details on the spine

- Stats NZ prototype spine paper
- Stats NZ linking methodology paper
- VHIN spine explainer

>>> Probabilistic linking errors

Probabilistic linking produces a unique identifier, snz_uid.
The snz_uid can then act as a primary/foreign key pair.

* Just because two records/interactions are linked, doesn't mean they belong to the same individual.
  This is a false positive.

>>> Probabilistic linking errors

Probabilistic linking produces a unique identifier, snz_uid. The snz_uid can then act as a primary/foreign key pair.

- Just because two records/interactions are linked, doesn't mean they belong to the same individual.
  This is a false positive.

- Just because two records/interactions are *not* linked, doesn't mean they *don't* belong to the same individual.
  This is a false negative.

>>> Probabilistic linking errors

Probabilistic linking produces a unique identifier, snz_uid.
The snz_uid can then act as a primary/foreign key pair.

- Just because two records/interactions are linked, doesn't mean they belong to the same individual.

  This is a false positive.

- Just because two records/interactions are *not* linked, doesn't mean they *don't* belong to the same individual.

  This is a false negative.

- Just because two records from different refreshes have the same snz_uid, definitely doesn't mean they have belong to the same individual.

  This is a silly mistake (IDI_Clean_20210820).

>>> Precision rate

The precision rate is the proportion of correct links, out of all links made. You can think of it as the probability that a randomly chosen link is correct. You can get the false positive rate from this as:

$$1 - (\texttt{precision rate}).$$

>>> Precision rate

The precision rate is the proportion of correct links, out of all links made. You can think of it as the probability that a randomly chosen link is correct. You can get the false positive rate from this as:

$$1 - (\texttt{precision rate}).$$

Stats NZ measures the precision rate, usually via clerical reviews of random samples of the links.

>>> Precision rate

The precision rate is the proportion of correct links, out of all links made. You can think of it as the probability that a randomly chosen link is correct. You can get the false positive rate from this as:

$$1 - (\text{precision rate}).$$

Stats NZ measures the precision rate, usually via clerical reviews of random samples of the links.

The priority of Stats NZ is to achieve a high precision rate. This involves a trade-off, with a *higher false negative rate*.

>>> Linkage bias

Linkage bias examines variables where the false negative rate is particularly high. For example, bias in year of birth is expected since older people have lived through longer periods of poor coverage, creating a linking bias. However, it is not easy to look at linked records versus records that didn't link, so estimating linkage bias is difficult.

>>> One-to-one relationship

When linkage is created between a schema and the spine, Stats NZ refers to it as a project.

"Each project ideally produces one-to-one links, where each record on one side links to at most one record on the other side. Duplicates are records which link to more than one record. How these are handled in the IDI depends on the projects."

>>> More on IDI_Clean

Individual level data:

* Many columns have snz_uid to link individuals

>>> More on IDI_Clean

Individual level data:

- Many columns have snz_uid to link individuals
- Schema names are like:
    - acc_clean (Accident Compensation Corporation)
    - dia_clean (Dept of Internal Affairs)
    - etc.

>>> More on IDI_Clean

Individual level data:

- Many columns have snz_uid to link individuals
- Schema names are like:
    - acc_clean (Accident Compensation Corporation)
    - dia_clean (Dept of Internal Affairs)
    - etc.
- Some wide tables: 700 columns in gss_clean.gss_person.

>>> More on IDI_Clean

Individual level data:

- Many columns have snz_uid to link individuals
- Schema names are like:
    - acc_clean (Accident Compensation Corporation)
    - dia_clean (Dept of Internal Affairs)
    - etc.
- Some wide tables: 700 columns in gss_clean.gss_person.
- The data schema:
    - Core info on individuals, available to all users
    - data.personal_detail (snz_in_spine)
    - data.snz_res_pop (resident on 30th June each year)
    - etc.

>>> More on IDI_Clean

Individual level data:

- Many columns have snz_uid to link individuals
- Schema names are like:
    - acc_clean (Accident Compensation Corporation)
    - dia_clean (Dept of Internal Affairs)
    - etc.
- Some wide tables: 700 columns in gss_clean.gss_person.
- The data schema:
    - Core info on individuals, available to all users
    - data.personal_detail (snz_in_spine)
    - data.snz_res_pop (resident on 30th June each year)
    - etc.
- The security schema:
    - Information related to linkage process.
    - security.concordance (link e.g., snz_uid to snz_acc_uid)

>>> A quick superpower

This code is T-SQL only:

```sql
USE IDI_Clean;
SELECT *
FROM Information_Schema.Columns
WHERE Table_Catalog = 'IDI_Clean';
```

>>> More on IDI_Metadata

Non-individual level data:

* Contains one schema, clean_read_CLASSIFICATIONS
* Contains around 300 tables

>>> More on IDI_Metadata

Non-individual level data:

* Contains one schema, clean_read_CLASSIFICATIONS
* Contains around 300 tables
* Translate codes, e.g.,
    * post_code
    * sla_ethnic_code
    * cyf_ethnicity_code
    * cor_ethnicity_code
    * cen_ethnic05
    * etc. (preserving 'full granularity')

>>> More on IDI_Metadata

Non-individual level data:

- Contains one schema, clean_read_CLASSIFICATIONS
- Contains around 300 tables
- Translate codes, e.g.,
  - post_code
  - sla_ethnic_code
  - cyf_ethnicity_code
  - cor_ethnicity_code
  - cen_ethnic05
  - etc. (preserving 'full granularity')
- Facts (e.g., socio-economic deprivation by meshblock)

>>> More on IDI_Metadata

Non-individual level data:

* Contains one schema, clean_read_CLASSIFICATIONS
* Contains around 300 tables
* Translate codes, e.g.,
    * post_code
    * sla_ethnic_code
    * cyf_ethnicity_code
    * cor_ethnicity_code
    * cen_ethnic05
    * etc. (preserving 'full granularity')
* Facts (e.g., socio-economic deprivation by meshblock)
* ''Uncommunicated changes sometimes happen''
* ''Almost certainly sub-optimal''
* ''Legacy code still doesn't use it''

>>> My favourite links

- Stats NZ paper - Use of the IDI
  Really good! "The first part of this report sets out to describe, from a researcher's point of view, what data is available, how it is structured, and the analytical platforms that are available". Much more!

- VHIN guides to getting started
  Very beginner friendly.

- Visual summary of available data as at July 2021.

- Current StatsNZ website
  Some material, no longer contains data dictionaries.

>>> From paper: Use of The IDI

* Most analyses are frequency counts and cross tabs.
* SAS dominates usage, but SQL is really the foundation.
* Need better ways to jointly develop code, facilitated by version control.
* Need more of the data joining, re-coding and filtering code written in SQL and executed on the database server.

>>> From paper: Use of The IDI

There is more instability in the database design than is necessary to accommodate real world change. Table and column names and types change; and some of the dimension reference data is in a separate database to the main data, when it exists in a database at all... Adding new data, including of a standard repeat type (e.g., a new survey) requires design work, not just new values in fact and dimensions. Because of this instability and the lack of code version control, researchers cannot be sure to reproduce any particular piece of data. Because of these environment limitations, analytical programs contain numerous ad hoc, context-specific and time-specific work-arounds and there is no legacy code base for reproducible research.

>>> Primary and foreign key pairs?

No foreign keys, generally

Let's look at some data dictionaries (click here)

>>> Primary and foreign key pairs?

No foreign keys, generally

Let's look at some data dictionaries (click here)

ACC Injury Claims Data: Serious Injury

* snz_uid

  Global, refreshed, not unique in table

* snz_acc_uid

* snz_acc_claim_uid

>>> Primary and foreign key pairs?

No foreign keys, generally

Let's look at some data dictionaries (click here)

ACC Injury Claims Data: Serious Injury

* snz_uid
  Global, refreshed, not unique in table
* snz_acc_uid
  Local, not refreshed, not unique in table
* snz_acc_claim_uid

>>> Primary and foreign key pairs?

No foreign keys, generally

Let's look at some data dictionaries (click here)

ACC Injury Claims Data: Serious Injury

* snz_uid
  Global, refreshed, not unique in table
* snz_acc_uid
  Local, not refreshed, not unique in table
* snz_acc_claim_uid
  Local, 'event ID', unique in table?

>>> Simplifying messy code

A Ministry of Health query...

```
SELECT year(IDI_Clean_20181020.moh_clean.
PRIMHD.moh_mhd_activity_start_date)
AS StartYear,
IDI_Clean_20181020.moh_clean.PRIMHD.snz_moh_uid
FROMIDI_Clean_20181020.moh_clean.PRIMHD
WHERE IDI_Clean_20181020.moh_clean.
PRIMHD.moh_mhd_activity_type_code !='T35'
GROUP BY year(IDI_Clean_20181020.moh_clean.
PRIMHD.moh_mhd_activity_start_date),
IDI_Clean_20181020.moh_clean.PRIMHD.snz_moh_uid
ORDER BY
year(IDI_Clean_20181020.moh_clean.PRIMHD
.moh_mhd_activity_start_date)
```

>>> Where are we now?

Day 3

1. Recap of days 1 and 2
2. Expanding the toolkit
3. Creating data
4. Working with CSVs
5. The IDI
6. Independent development
7. Windowing part 2

   Send me questions and give feedback

Page is hyperlinked: click a topic above to jump to it.

>>> Independent development

Live walk-through using StackExchange database.

>>> Where are we now?

Day 3

1. Recap of days 1 and 2
2. Expanding the toolkit
3. Creating data
4. Working with CSVs
5. The IDI
6. Independent development
7. Windowing part 2

   Send me questions and give feedback

Page is hyperlinked: click a topic above to jump to it.

>>> Now, the actual window functions

Function	Returns
FIRST_VALUE	Entry in first row of partition
LAST_VALUE	Entry in last row of partition
LAG	Entry one row behind current row
LEAD	Entry one row ahead of current row
ROW_NUMBER	Number of current row in partition
RANK	Rank of current row in partition
DENSE_RANK	Rank, without gaps
PERCENT_RANK	Percentage of rank value
CUME_DIST	Cumulative distribution value
NTILE	Bucket numbers (like histogram)

All potentially return more than one value per partition.

We will explore them live in a moment, but first...

>>> What is a rank?

A six day sausage sizzle, starting on NYE, 1999

SausageSizzleSummary

SaleDate	Product	Sales
1999-12-31	pork	3
1999-12-31	veggie	3
2000-01-01	pork	2
2000-01-01	veggie	7
2000-01-02	pork	6
2000-01-02	veggie	6
2000-01-03	pork	6
2000-01-03	veggie	2
2000-01-04	pork	1
2000-01-05	veggie	5

>>> What is a rank?

RESULT			
Sales	row_number	rank	dense_rank
1	1	1	1
2	2	2	2
2	3	2	2
3	4	4	3
3	5	4	3
5	6	6	4
6	7	7	5
6	8	7	5
6	9	7	5
7	10	10	6

>>> What is a rank?

RESULT			
Sales	row_number	rank	dense_rank
1	1	1	1
2	2	2	2
2	3	2	2
3	4	4	3
3	5	4	3
5	6	6	4
6	7	7	5
6	8	7	5
6	9	7	5
7	10	10	6

Returns the order of the rows

>>> What is a rank?

RESULT			
Sales	row_number	rank	dense_rank
1	1	1	1
2	2	2	2
2	3	2	2
3	4	4	3
3	5	4	3
5	6	6	4
6	7	7	5
6	8	7	5
6	9	7	5
7	10	10	6

Rank identifies ties, and may skip over values

>>> What is a rank?

RESULT

Sales	row_number	rank	dense_rank
1	1	1	1
2	2	2	2
2	3	2	2
3	4	4	3
3	5	4	3
5	6	6	4
6	7	7	5
6	8	7	5
6	9	7	5
7	10	10	6

Dense rank: a rank that doesn't skip values

>>> ORDER BY inside OVER

On the previous slide, the order of Sales was important.

```
SELECT Sales,
      ROW_NUMBER() OVER(ORDER BY Sales) AS row_num_sales,
      RANK()       OVER(ORDER BY Sales) AS rank_sales,
      DENSE_RANK() OVER(ORDER BY Sales) AS dense_rank_sales
FROM SausageSizzleSummary;
```

All of the window functions are generally used with an ORDER BY clause.

>>> ORDER BY inside OVER

On the previous slide, the order of Sales was important.

```
SELECT Sales,
      ROW_NUMBER() OVER(ORDER BY Sales) AS row_num_sales,
      RANK()       OVER(ORDER BY Sales) AS rank_sales,
      DENSE_RANK() OVER(ORDER BY Sales) AS dense_rank_sales
FROM SausageSizzleSummary;
```

All of the window functions are generally used with an ORDER BY clause.

Now for a live demo.

>>> Exercises

Practice makes perfect.

Click here to find the textbook.