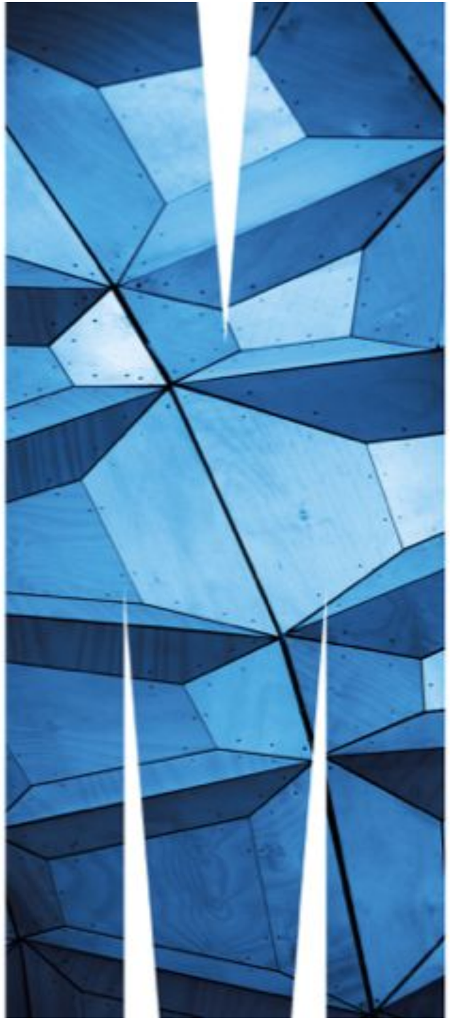
MONASH University

MONASH
INFORMATION
TECHNOLOGY



FIT2094 - FIT3171 Databases

Week 3 - The Relational Database Model

Workshop S1 2022

Overview

Once we have a conceptual model, it is time to move to the second stage and map this to a logical model

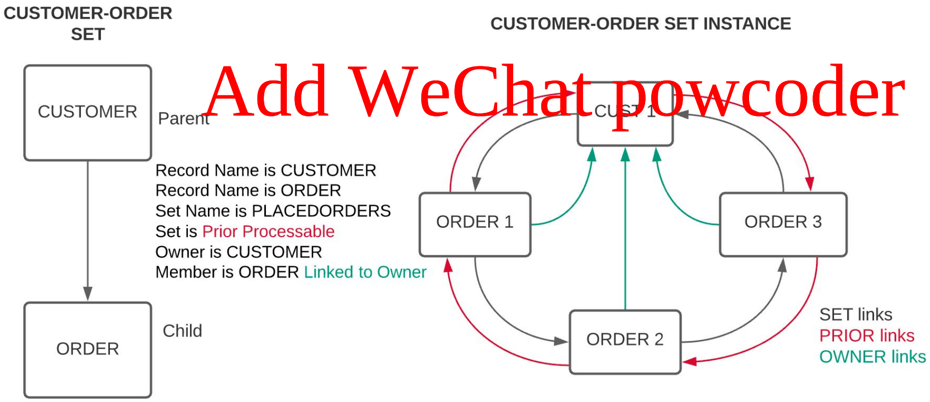
For our unit this will involve mapping to the *Relational Model* in preparation for implementation in a RDBMS. First before we consider this mapping it is necessary to have a clear understanding of the Relation Model and it use:

- Relational Model
- Relational Algebra

Early Database Models

- Hierarchical (1970's eg. IBM Information Management System (IMS))
  - 1:M relationships, a tree of linked records, child has only one parent
- Network (1970's eg. Integrated Data Store IDS, basis for the CODASYL group)
  - child may have multiple parents
- Both Navigational - move around in data via *embedded links* (pointers)

Network:



The Relational Model

- Introduced by CODD in 1970 - the fundamental basis for the relational DBMS
- Basic structure is the mathematical concept of a RELATION mapped to the 'concept' of a table (tabular representation of relation)
  - Relation - abstract object
  - Table - pictorial representation
  - Storage structure - "real thing" - eg. isam file of 1's and 0's
- Relational Model Terminology
  - DOMAIN - set of atomic (indivisible) values
  - Examples (name, data type, data format):
    - customer_number domain - 5 character string of the form xxxdd
    - name domain - 20 character string
    - address domain - 30 character string containing street, town & postcode
    - credit_limit domain - money in the range $1,000 to $99,999

A Relation

- A relation consists of two parts
  - heading
  - body
- **Relation Heading**
  - Also called Relational **Schema** consists of a fixed set of attributes
    - R (A1,A2,.......An)
      - R = relation name, Ai = attribute i
  - Each attribute corresponds to one underlying domain:
    - Customer relation heading:
      - CUSTOMER (custno, custname, custadd, custcredlimit)
        - » dom(custno) = customer_number
        - » dom(custname) = name
        - » dom(custadd) = address
        - » dom(custcredlimit) = credit_limit

custno	custname	custadd	custcredlimit

Relation Body

- **Relation Body**
  - Also called Relation Instance (state of the relation at any point in time)
    - r(R) = {t1, t2, t3, ..., tm}
    - consists of a time-varying set of n-tuples
      - Relation R consists of tuples t1, t2, t3 .. tm
      - m = number of tuples = **relation cardinality**

    - each n-tuple is an ordered list of n values
    - t = < v1, v2, ..., vn>
      - n = number of values in tuple (no of attributes) = **relation degree**

  - In the tabular representation:
    - Relation heading ➪ column headings
    - Relation body ➪ set of data rows

custno	custname	custadd	custcredlimit
SMI13	SMITH	Wide Rd, Clayton, 3168	2000
JON44	JONES	Narrow St, Clayton, 3168	10000
BRO23	BROWN	Here Rd, Clayton, 3168	10000

Relation Properties

- **No duplicate tuples**
  - by definition sets do not contain duplicate elements
    - hence tuples must be unique
- **Tuples are unordered within a relation**
  - by definition sets are not ordered
    - hence tuples can only be accessed by content
- **No ordering of attributes within a tuple**
  - by definition sets are not ordered

Relation Properties cont'd

- **Tuple values are atomic** - cannot be divided
  - EMPLOYEE (eid, ename, departno, dependants)
    - not allowed: dependants (depname, depage) multivalued
  - hence no multivalued (repeating) attributes allowed, called the first normal form rule
- COMPARE with tabular representation
  - normally nothing to prevent duplicate rows
  - rows are ordered
  - columns are ordered
  - tables and relations are not the same 'thing'

Functional Dependency

- **Functional Dependency**:
  - A set of attributes A functionally determines an attribute B if, and only if, for each A value, there is exactly one value of B in the relation. It is denoted as A → B (A determines B, or B depends on A)
    - orderno → orderdate
    - prodno → proddesc
    - orderno, prodno → qtyordered

ORDERNO	ORDERDATE
10	01/MAY/19
11	02/MAY/19
12	03/MAY/19
13	04/MAY/19
14	04/MAY/19
15	05/MAY/19
16	06/MAY/19

ORDERNO	PRODNO	QTYORDERED	LINEPRICE
10	101	1	11.98
11	101	1	11.98
11	103	2	123.58
12	104	10	479.8
13	105	2	140.36
14	106	1	31.99
15	107	3	116.73

PRODNO	PRODDESC	PRODUNITPRICE
101	Salmon - Smoked, Sliced	11.98
102	Brocolinni - Gaylan, Chinese	80.75
103	Pasta - Lasagne, Fresh	61.79
104	Melon - Cantaloupe	47.98
105	Wine - Peller Estates Late	70.18
106	Peas - Pigeon, Dry	31.99
107	Pumpkin - Seed	38.91

Relational Model Keys

- A **superkey** of a relation R is an attribute or set of attributes which exhibits only the uniqueness property
  - No two tuples of R have the same value for the superkey (Uniqueness property)
  - t1[superkey] ≠ t2[superkey]

- A **candidate key** CK of a relation R is an attribute or set of attributes which exhibits the following properties:
  - Uniqueness property (as above), *and*
  - No proper subset of CK has the uniqueness property (Minimality or Irreducibility property) ie. a minimal superkey

- One candidate key is chosen to be the **primary key** (PK) of a relation. Remaining candidate keys are termed alternate keys (AK).

Many possible superkeys

Potentially many possible candidate keys

Only ONE primary key (may be composed of many attributes - a composite primary key)

Free text answer in Flux (after Group discussion):

Q1. List all the super keys for:

Unit code	Mark	StudId	Year
FIT2094	80	111	2016
FIT2094	20	111	2015
FIT2004	100	111	2016
FIT2004	40	222	2015
FIT2004	40	333	2015

Notes:

- treat this as only a small sample of the data
- the attributes are fixed.

Selection of a Primary key

- **A primary key must be chosen considering the data that *may be added to the table in the future***
  - Names, dates of birth etc are rarely unique and as such are not a good option
  - PK should be free of 'extra' semantic meaning and security compliant, preferably a single attribute, preferably numeric (see Table 5.3 Coronel & Morris)
  - Natural vs Surrogate primary key
    - ENROLMENT(unitcode, student_id, enrol_sem, enrol_year, enrol_mark, enrol_grade)
      - Superkey
      - CK
      - PK
      - Issues with PK?

TABLE 5.3

DESIRABLE PRIMARY KEY CHARACTERISTICS

PK CHARACTERISTIC	RATIONALE
Unique values	The PK must uniquely identify each entity instance. A primary key must be able to guarantee unique values. It cannot contain nulls.
Nonintelligent	The PK should not have embedded semantic meaning other than to uniquely identify each entity instance. An attribute with embedded semantic meaning is probably better used as a descriptive characteristic of the entity than as an identifier. For example, a student ID of 650973 would be preferred over Smith, Martha L. as a primary key identifier.
No change over time	If an attribute has semantic meaning, it might be subject to updates, which is why names do not make good primary keys. If Vickie Smith is the primary key, what happens if she changes her name when she gets married? If a primary key is subject to change, the foreign key values must be updated, thus adding to the database work load. Furthermore, changing a primary key value means that you are basically changing the identity of an entity. In short, the PK should be permanent and unchangeable.
Preferably single-attribute	A primary key should have the minimum number of attributes possible (irreducible). Single-attribute primary keys are desirable but not required. Single-attribute primary keys simplify the implementation of foreign keys. Having multiple-attribute primary keys can cause primary keys of related entities to grow through the possible addition of many attributes, thus adding to the database workload and making (application) coding more cumbersome.
Preferably numeric	Unique values can be better managed when they are numeric, because the database can use internal routines to implement a counter-style attribute that automatically increments values with the addition of each new row. In fact, most database systems include the ability to use special constructs, such as Autonumber in Microsoft Access, sequence in Oracle, or uniqueidentifier in MS SQL Server to support self-incrementing primary key attributes.
Security-compliant	The selected primary key must not be composed of any attribute(s) that might be considered a security risk or violation. For example, using a Social Security number as a PK in an EMPLOYEE table is not a good idea.

Null in the Relational Model *Implementation*

- ***NULL is a concept <u>created</u> and <u>implemented by SQL</u>, does not exist in classical relational algebra***
- NULL is NOT a value - is a representation of the fact that there is *NO VALUE*
- Reasons for a NULL:
  - VALUE NOT APPLICABLE -
    - EMP relation - empno, deptno, salary, commission
      - commission only applies to staff in sales dept
  - VALUE UNKNOWN -
    - Joe's salary is NULL, Joe's salary is currently unknown
  - VALUE DOES NOT EXIST -
    - Tax File Number - is applicable to all employees but Joe may not have a number at this time
  - VALUE UNDEFINED -
    - Certain items explicitly undefined eg. divide by zero
      - Columns Number_of_payments, Total_payments
      - Column Average_payment_made
      - If Number_of_payments = 0 => Average undefined

Writing Relations

- Relations may be represented using the following notation:
  - RELATION_NAME (attribute1, attribute2,…)
- Relation_name must not be pluralised (is a set name)
- The primary key is underlined
- Example:
  - **STAFF (staff_id, staff_surname, staff_initials, staff_address, staff_phone)**

Q2. A well designed relational database (a database based on the relational model) has:

A. No redundant data
B. Minimal redundant data
C. A large amount of redundant data
D. A level of redundancy based on the vendors implementation

Relational Database

- A relational database is a collection of normalised relations.
- Normalisation is part of the design phase of the database and will be discussed in a later lecture.

Example relational database:

ORDER (<u>order_id</u>, order_date)

ORDER_LINE (<u>order_id</u>, <u>prod_id</u>, ol_quantity)

PRODUCT (<u>prod_id</u>, prod_desc, prod_unitprice)

Foreign Key (FK) - Implementation

- FK: An attribute/s in a relation that exists in the same, or another relation as a Primary Key.
- Referential Integrity
  - **A Foreign Key value must either *match* the *full primary key* in a relation or be NULL.**
- The pairing of PK and FK creates relationships (**logical** connections) between tables when implemented in a RDBMS. Hence the abstraction away from the underlying storage model.

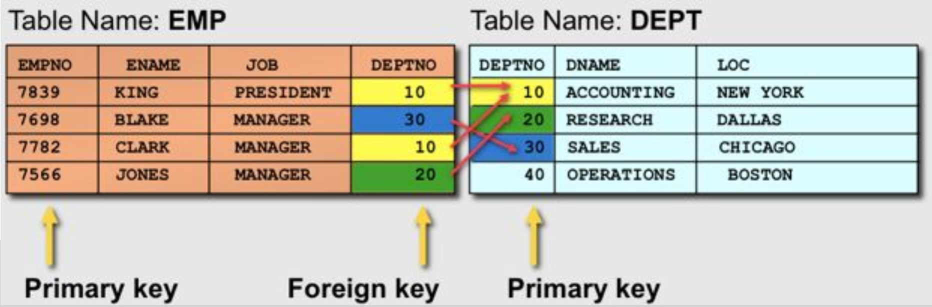


Table Name: **EMP**

EMPNO	ENAME	JOB	DEPTNO
7839	KING	PRESIDENT	10
7698	BLAKE	MANAGER	30
7782	CLARK	MANAGER	10
7566	JONES	MANAGER	20

Table Name: **DEPT**

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

Q4. Business rules:
Runners may form a team, the runner who registers the team is recorded as the team leader. Each team can have up to 5 members (runners).

Identify the FK(s):

TEAM(<u>team_id</u>, team_name, team_leader)
RUNNER(<u>runner_id</u>, runner_name, team_id)

A. team_leader in TEAM
B. team_id in TEAM
C. runner_id in RUNNER
D. team_id in RUNNER

Data Integrity - Implementation

- Entity integrity
  - Primary key value must not be NULL.
    - No duplicate tuple property then ensures that each primary key must be unique
    - Implemented in the RDBMS via a unique index on the PK
- Referential integrity
  - The values of FK must either match a value of a full PK in the related relation or be NULL.
- Column/Domain integrity
  - All values in a given column must come from the same domain (the same data type and range).

Q5. The following set of relations:

HOSPITAL (<u>hosp_id</u>, hosp_name, hosp_phone)
DOCTOR (<u>hosp_id</u>, <u>dr_id</u>, dr_name, dr_mobile)
PATIENT (pat_id, pat_name, pat_dob, dr_id)

A. have no integrity issues
B. violate entity integrity
C. violate referential integrity
D. violate column/domain integrity

Multiple responses allowed

Relational DMLs

- Relational Calculus
- Relational Algebra
- Transform Oriented Languages (e.g. SQL)
- Graphical Languages
- Exhibit the “closure” property - queries on relations produce relations

Relational Calculus

- Based on mathematical logic.
- Non-procedural.
- Primarily of theoretical importance.
- May be used as a yardstick for measuring the power of other relational languages (“relational completeness”).
- Operators may be applied to any number of relations.

RELATIONAL ALGEBRA

Manipulation of relational data

Relational Algebra

- Relationally complete.
- Procedural.
- Operators only apply to at most two relations at a time.
- 8 basic operations:
  - *single relation:* selection, projection
  - *two relations:*
    - cartesian product, join
    - union
    - intersection
    - difference
    - division
- *Standard RA/pure form has no concept of NULL (Databases units use standard RA)*

Relational Operation PROJECT

PRDETAIL (<u>project_code</u>, project_manager, project_bid_price)

π

PROJECT_CODE	PROJECT_MANAGER	PROJECT_BID_PRICE
21-5Z	Holly B. Parker	$16,833,460.00
25-2D	Jane D. Grant	$12,500,000.00
25-5A	George F. Dorts	$32,512,420.00
25-9T	Holly B. Parker	$21,563,234.00
27-4Q	George F. Dorts	$10,314,545.00
29-2D	Holly B. Parker	$25,559,999.00
31-7P	William K. Moor	$56,850,000.00

How many tuples?
How many attributes?

Show all project manager:

$$\text{RESULT} = \pi_{\text{project\_manager}}\ \text{PRDETAIL}$$

Relational Operation SELECT

PRDETAIL (<u>project_code</u>, project_manager, project_bid_price)

σ

PROJECT_CODE	PROJECT_MANAGER	PROJECT_BID_PRICE
21-5Z	Holly B. Parker	$16,833,460.00
25-2D	Jane D. Grant	$12,500,000.00
25-5A	George F. Dorts	[illegible]
25-9T	Holly B. Parker	$21,563,234.00
27-4Q	George F. Dorts	$10,314,545.00
29-2D	Holly B. Parker	[illegible]
31-7P	William K. Moor	$56,850,000.00

Show details of project 25-5A:

How many tuples?
How many attributes?

$$\text{RESULT} = \sigma_{\text{project\_code} = \text{25-5A}}\ \text{PRDETAIL}$$

Relational Operation Multiple Actions

PRDETAIL (<u>project_code</u>, project_manager, project_bid_price)

2

1

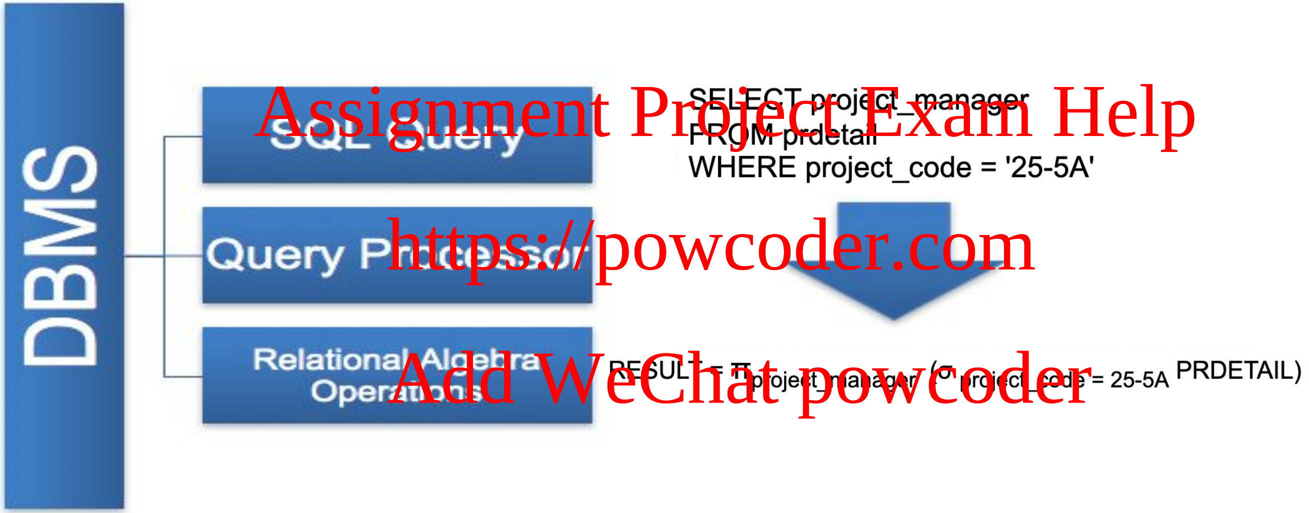
PROJECT_CODE	PROJECT_MANAGER	PROJECT_BID_PRICE
21-5Z	Holly B. Parker	$16,833,460.00
25-2D	Jane D. Grant	$12,500,000.00
25-5A	George F. Dorts	$32,512,420.00
25-9T	Holly B. Parker	$21,563,234.00
27-4Q	George F. Dorts	$10,314,545.00
29-2D	Holly B. Parker	$25,559,999.00
31-7P	William K. Moor	$56,850,000.00

How many tuples?
How many attributes?

Show the project manager of project 25-5A

RESULT = $\pi_{project\_manager}$ ($\sigma_{project\_code\ =\ 25\text{-}5A}$ PRDETAIL)

SQL vs Relational Algebra in the Database



Q5. Relational Algebra *select* and *project*

The following relations represent a karate dojo member training attendance:

SENSEI (sensei_id, sensei_name)
TRAINING_SCHEDULE (training_day, training_time, group_id, sensei_id)
ATTENDANCE (training_day, training_time, member_id, attendance_date)
MEMBER (member_id, member_name, member_dob, member_belt, group_id)
GROUP (group_id, group_name, group_age_range)

A. Primary keys are underlined

B. A karate member falls into one of the age level groups Tiny Tiger (for 4-7 year old), Young Dragon (for 8-14 years old), or Adult (for 14+ years old) and owns a certain color of belt (e.g. white, green, brown or black)

C. Sensei (Karate teachers) are scheduled to train an age level group of karate members in a particular day and time (e.g. Sensei Luke Nakamura trains Tiny Tiger members every Tuesday 5pm)

D. A karate member may attend more than one training schedule of their age level group in a given week.

Write the relational algebra for the following query (**your answer must show an understanding of query efficiency**):

(1) Show the name and dob of all black belt members.

JOIN

- Join operator used to combine data from two or more relations, based on a common attribute or attributes.
- Different types:
  - theta-join
  - equi-join
  - natural join

THETA JOIN (Generalised join)

$$(\text{Relation\_1}) \bowtie_F (\text{Relation\_2})$$

- *F* is a predicate (i.e. truth-valued function) which is of the form Relation_1.a_i θ Relation2.b_i
  - CUSTOMER.cust_no θ ORDER.cust_no
- θ is one of the standard arithmetic comparison operators, $<, \leq, =, \geq, >$
- Most commonly, θ is equals (=), but can be *any* of the operators
  - EMPLOYEE.emp_sal > SALARYSCALE.step_5

NATURAL JOIN

STUDENT

studid	studname
1	Alice
2	Bob

MARK

studid	unitcode	mark
1	1004	95
2	1045	55
1	1045	90

Step 1: STUDENT X MARK

Step 2: delete rows where IDs do not match (select =)

Result at Step 2 is an Equijoin (STUDENT $\bowtie_{\text{(STUDENT.studid = MARK.studid)}}$ MARK)

STUDENT.studid	studname	MARK.studid	unitcode	mark
1	Alice	1	1004	95
~~1~~	~~Alice~~	~~2~~	~~1045~~	~~55~~
1	Alice	1	1045	90
~~2~~	~~Bob~~	~~1~~	~~1004~~	~~95~~
2	Bob	2	1045	55
~~2~~	~~Bob~~	~~1~~	~~1045~~	~~90~~

NATURAL JOIN

STUDENT

studid	studname
1	Alice
2	Bob

MARK

studid	unitcode	mark
1	1004	95
2	1045	55
1	1045	90

Step 1: STUDENT X MARK
Step 2: delete rows where IDs do not match (select =)
Step 3: delete duplicate columns (project away)

Result at Step 3 is a Natural Join

STUDENT.studid	studname	MARK.studid	unitcode	mark
1	Alice	1	1004	95
1	Alice	1	1045	90
2	Bob	2	1045	55

NATURAL JOIN

STUDENT

studid	studname
1	Alice
2	Bob

$\bowtie$

MARK

studid	unitcode	mark
1	1004	95
2	1045	55
1	1045	90

Step 1: STUDENT X MARK
Step 2: delete rows where IDs do not match (select =)
Step 3: delete duplicate columns (project away)

studid	studname	unitcode	mark
1	Alice	1004	95
1	Alice	1045	90
2	Bob	1045	55

A natural join of STUDENT and MARK

Q6. Which of the following statements returns a natural join of the two relations on the agent code (agent_cd and agent_code)?

Relation : CUSTOMER

CUS_CODE	CUS_LNAME	CUS_ZIP	AGENT_CD
1132445	Walker	32145	231
1217782	Adares	32145	125
1312243	Rakowski	34129	167
1321242	Rodriguez	37134	125
1542311	Smithson	37134	333
1657399	Vanloo	32145	231

Relation : AGENT

AGENT_CODE	AGENT_PHONE
125	6152439887
167	6153426778
231	6152431124
333	9041234445

A. $\sigma_{\text{agent\_cd = agent\_code}}$ (CUSTOMER X AGENT)

B. $\pi_{\text{cus\_code, cus\_lname, cus\_zip, agent\_code, agent\_phone}}$ ($\sigma_{\text{agent\_cd = agent\_code}}$ (CUSTOMER X AGENT))

C. $\sigma_{\text{agent\_cd = agent\_code}}$ ($\pi_{\text{cus\_code, cus\_lname, cus\_zip, agent\_code, agent\_phone}}$ (CUSTOMER X AGENT))

D. All of the above

E. None of the above

UNION, INTERSECT, DIFFERENCE

STOREA

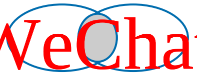
product_id	product_name
1	LG Nano91 75" 4K
2	TCL P725 65" 4K UHD
3	Sony X85J 75" Bravia

UNION (STOREA ∪ STOREB)

product_id	product_name
1	LG Nano91 75" 4K
2	TCL P725 65" 4K UHD
3	Sony X85J 75" Bravia
33	LG C1 48" Self Lit OLED 4K

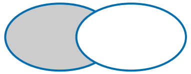
STOREB

product_id	product_name
1	LG Nano91 75" 4K
2	TCL P725 65" 4K UHD
33	LG C1 48" Self Lit OLED 4K



INTERSECT (STOREA ∩ STOREB)

product_id	product_name
1	LG Nano91 75" 4K
2	TCL P725 65" 4K UHD



DIFFERENCE (STOREA - STOREB)

product_id	product_name
3	Sony X85J 75" Bravia

Union compatible relations required

Q7. Relational Algebra

The following relations represent a karate dojo member training attendance:

SENSEI (sensei_id, sensei_name)
TRAINING_SCHEDULE (training_day, training_time, group_id, sensei_id)
ATTENDANCE (training_day, training_time, member_id, attendance_date)
MEMBER (member_id, member_name, member_dob, member_belt, group_id)
GROUP (group_id, group_name, group_age_range)

A. Primary keys are underlined
B. A karate member falls into one of the age level groups: Tiny Tiger (for 4-7 year old), Young Dragon (for 8-14 years old), or Adult (for 14+ years old) and owns a certain color of belt (e.g. white, green, brown or black)
C. Sensei (Karate teachers) are scheduled to train an age level group of karate members in a particular day and time (e.g. Sensei Luke Nakamura trains Tiny Tiger members every Tuesday 5pm)
D. A karate member may attend more than one training schedule of their age level group in a given week.

Write the relational algebra for the following query (**your answer must show an understanding of query efficiency**):

(2) Show the name, belt colour and attendance dates of the member with an id of 12345

ANSWER Q7

(2) Show the name, belt colour and attendance dates of the member with an id of 12345

R2 = π member_name, member_belt, attendance_date (σ member_id = 12345 (MEMBER $\bowtie$ ATTENDANCE))

- this is the CANONICAL QUERY - not technically incorrect, but very inefficient, say member 12345 has only attended once in say 1000 tuples in ATTENDANCE. The join between MEMBER and ATTENDANCE yields, in such a scenario, 1000 tuples, 999 of which are unnecessary.

Your solution must demonstrate an understanding of efficiency:

A2a = π member_id, attendance_date (σ member_id = 12345 ATTENDANCE)

A2b = π member_id, member_name, member_belt (σ member_id = 12345 MEMBER)

R2 = π member_name, member_belt, attendance_date (A2a $\bowtie$ A2b)

Q8. Relational Algebra POST WORKSHOP TASK - answer available Sunday 5PM

The following relations represent a karate dojo member training attendance:

SENSEI (sensei_id, sensei_name)
TRAINING_SCHEDULE (training_day, training_time, group_id, sensei_id)
ATTENDANCE (training_day, training_time, member_id, attendance_date)
MEMBER (member_id, member_name, member_dob, member_belt, group_id)
GROUP (group_id, group_name, group_age_range)

A. Primary keys are underlined
B. A karate member falls into one of the age level groups: Tiny Tiger (for 4-7 year old), Young Dragon (for 8-14 years old), or Adult (for 14+ years old) and owns a certain color of belt (e.g. white, green, brown or black)
C. Sensei (Karate teachers) are scheduled to train an age level group of karate members in a particular day and time (e.g. Sensei Luke Nakamura trains Tiny Tiger members every Tuesday 5pm)
D. A karate member may attend more than one training schedule of their age level group in a given week.

Write the relational algebra for the following query (**your answer must show an understanding of query efficiency**):

(3) Show the id, name and age level group name of members who were absent (did not attend any training) between 01-03-2021 and 31-03-2021 (inclusive).