Databases Final Group Project: Realtek Inc.

Bintu Alkassoum, Zhiling (Catherine) Cao, Yi (Ethan) Wang, Xuechun (Lucia) Fu,

Pooja Ganiga, Jiawei (Richard) Geng, Somiya Jena, Tahniat Khan

Schulich School of Business: York University

MMAI 5100 Database Fundamentals

Dr. Altay Aksulu

December 12th, 2021

Case Description:

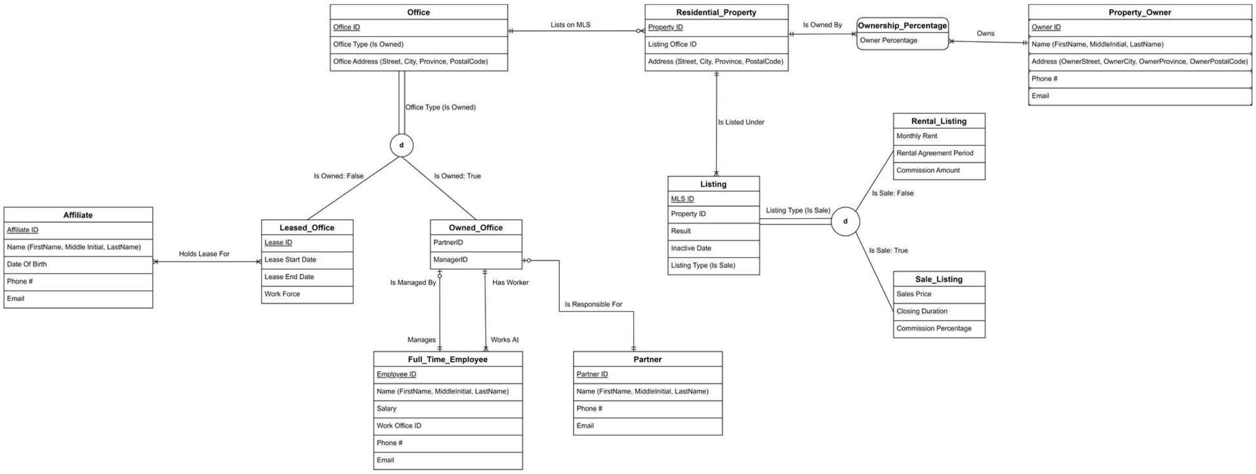
Realtek Inc. is a Canadian real estate property firm with offices located across Canada. Realtek Inc. assists their clients with both the purchasing, selling, and leasing of residential property. Each of these offices are owned by either Realtek Inc. or affiliated realtor(s). Additionally, Realtek is composed of various departments, each with distinct informational needs. As the real estate industry continues to grow across Canada, Realtek will need to implement a database system to achieve the following objectives:

1) Database Flexibility: This will ensure that the database is adaptable to changes in data volume and structure, allowing new information to be easily integrated;
2) Database Optimization: As Realtek continues to grow, optimization will guarantee that the database remains efficient, useful, and accurate;
3) Strong Usability: Realtek's database will need to be easily and conveniently accessible to a number of individuals with various roles and responsibilities within the company; as Realtek grows, intradepartmental roles will also likely expand.

Database Design:

We have decided that PostgreSQL RDBMS would be the preferred implementation platform.
A png file for each conceptual, logical and physical design is attached. The sql files for creating and populating tables are attached.
Please see the following database designs, along with their corresponding assumptions:

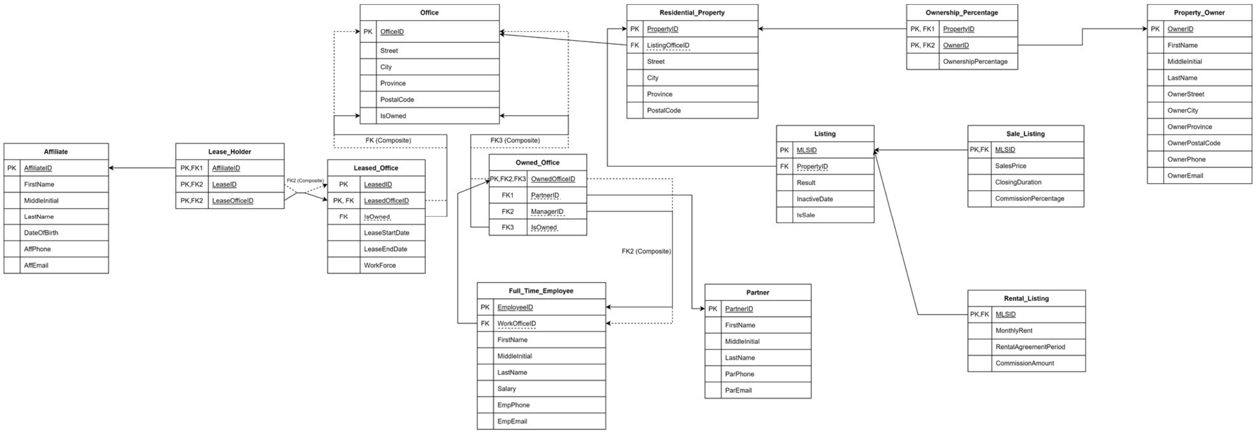
Conceptual Design:



For our conceptual design, we assume that:

- More partners than owned offices. Partners can exist on their own.
- We have Partner Names
- Partners don't belong to employees. They are owners.
- Affiliates cannot exist on their own. An affiliate must lease at least an office.
- All corporate employees are full time employees (FTEs).
- Full-time employees must work at one and only one office. (no remote working)
- Property owners cannot exist on their own. Property owners must own at least one property.

Logical Design:

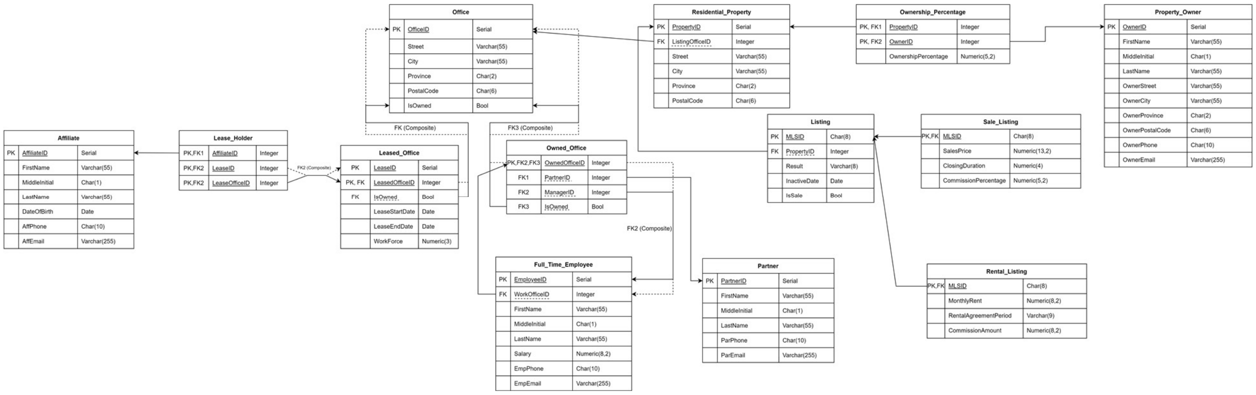


(Dotted lines are used to show references that are part of composite foreign keys)

For our logical design, we assume that:

- LeaseID is assigned for each renewal/beginning for each leased office.
- We are assuming that Affiliates are not likely to be FTE. If it happens, he/she will have both Affiliate ID and EmployeeID.
- We are assuming we have partner's Firstname, Middle Initial and Lastname.
- We are assuming we don't have employee details in leased offices but we have the number of employees (as workforce).
- We are assuming we gather the phone number and email for partners, affiliates, full-time employees, and property owners.

Physical Design:



(Dotted lines are used to show references that are part of composite foreign keys)

For our physical design, we assume that:

- Longest Street name in Canada is 31 characters. Plus space, plus street number, plus dash, plus unit number. we assume 55 is enough. (31(street name) + 5(street number) + 1(possible dash) + 4(unit number) + 5(spaces)).
- First or Last Names are less than 55 characters. (If longer, we use the first 55 characters).
- Only the first middle name will be used as Middle Name Initial.
- MLSID is in the form the Canadian MLS system and with the same format as in the MLS system.

Table Breakdown: domain definitions and assumptions for physical design:

Affiliate

	AffiliateID	FirstName	MiddleInitial	LastName	DateOfBirth	AffPhone	AffEmail
PK & FK constraint	PK						
Data Type	Serial	Varchar(55)	Char(1)	Varchar(55)	Date	Char(10)	Varchar(255)
Nullable	No	No	Yes	No	No	Yes	Yes
Unique	Yes	No	No	No	No	No	No
Display format	999	Xxxxx	X	Xxxxx	9999-99-99	1234567890	Xxx@xxxx.xxx
Range/valid entries			One letter			0000000000 to 9999999999	
Example	3	John	F	Doe	1990-08-01	23676589765	johndoe@gmail.com

Partner

	PartnerID	FirstName	MiddleInitial	LastName	ParPhone	ParEmail
PK & FK constraint	PK					
Data Type	Serial	Varchar(55)	Char(1)	Varchar(55)	Char(10)	Varchar(255)
Nullable	No	No	Yes	No	Yes	Yes
Unique	Yes	No	No	No	No	No
Display format	999	Xxxxx	X	Xxxxx	1234567890	Xxx@xxxx.xxx
Range/valid entries			One letter		0000000000 to 9999999999	
Example	101	John	F	Doe	23676589765	johndoe@gmail.com

Office

	OfficeID	Street	City	Province	PostalCode	IsOwned
PK & FK constraint	PK					
Data Type	Serial	Varchar(55)	Varchar(55)	Char(2)	Char(6)	Bool
Nullable	No	No	No	No	No	No
Unique	Yes	No	No	No	No	No
Display format	999	111 Xxxxx Xxxx	XXXXX	XX	X1X1X1	TRUE/FALSE
Range/valid entries						TRUE/FALSE
Example	101	49 Leslie St	Toronto	ON	M4M3C3	TRUE

Leased_Office

	LeaseID	LeasedOfficeID	IsOwned	LeaseStartDate	LeaseEndDate	WorkForce
PK & FK constraint	PK(composite)	PK(composite), FK(composite)	FK(composite)			
Data Type	Serial	Integer	Bool	Date	Date	Numeric(3)
Nullable	No	No	No	No	No	Yes
Unique	Yes	No	No	No	No	No
Display format	999	999	XXXXX	9999-99-99	9999-99-99	999
Range/valid entries			FALSE			0-999
Example	1	101	FALSE	1990-08-01	2020-08-01	12

Assumption for the above table Leased_Office:

Workforce in any leased office is less than 1000.

Note:

Added Attribute: IsOwned to enforce disjoint rule albeit it is redundant.

Used composite PK because of conversion from supertype/subtype relationship.

Lease_Holder

	AffiliateID	LeaseID	LeasedOfficeID
PK & FK constraint	PK (composite), FK1	PK (composite), FK2 (composite)	PK (composite), FK2 (composite)
Data Type	Integer	Integer	Integer
Nullable	No	No	No
Unique	No	No	No
Display format	999	999	999
Range/valid entries			
Example	1	1	1

Owned_Office

	OwnedOfficeID	PartnerID	IsOwned	ManagerID
PK & FK constraint	PK, FK2 (composite), FK3 (composite)	FK1	FK2 (composite)	FK3 (Composite)
Data Type	Integer	Integer	Integer	Bool
Nullable	No	No	No	Yes
Unique	Yes	No	No	No
Display format	999	999	XXXX	999
Range/valid entries			TRUE	
Example	1	101	TRUE	101

Assumption for the above table Owned_Office:

If owned_office is no longer owned, we will remove the office info from the database.

Note:

Added Attribute: IsOwned to enforce disjoint rule albeit it is redundant.

MangerID is nullable to enable inserting new offices.

Full_Time_Employee

	EmployeeID	FirstName	MiddleInitial	LastName	Salary	WorkOfficeID	EmpPhone	EmpEmail
PK & FK constraint	PK					FK		
Data Type	Serial	Varchar(55)	Char(1)	Varchar(55)	Numeric(8,2)	Integer	Char(10)	Varchar(255)
Nullable	No	No	Yes	No	No	No	Yes	Yes
Unique	Yes	No	No	No	No	No	No	No
Display format	999	Xxxxx	X	Xxxxx	9999.99	999	1234567890	Xxx@xxxx.xxx
Range/valid entries			One letter		0 to 999,999.99		0000000000 to 9999999999	
Example	3	John	F	Doe	6600.50	1	23676589765	johndoe@gmail.com

Assumption for the above table Full_Time_Employee:

Salary shown is the monthly salary and will be from 0 to 999,999.99.

Property_Owner

	OwnerID	FirstName	Middle Initial	LastName	OwnerStreet	OwnerCity	OwnerProvince	OwnerPostalCode	OwnerPhone	OwnerEmail
PK & FK constraint	PK									
Data Type	serial	Varchar(55)	Char(1)	Varchar(55)	Varchar(55)	Varchar(55)	Char(2)	Char(6)	Char(10)	Varchar(255)
Nullable	No	No	Yes	No	No	No	No	No	Yes	Yes
Uniqueness	Yes	No	No	No	No	No	No	No	No	No
Display format	999	Xxxxx	X	Xxxxx	111 Xxxxx Xxxx	XXXXX	XX	X1X1X1	1234567890	Xxx@xxxx.xxx
Range/valid entries			One letter						0000000000 to 9999999999	
Example	357	John	F	Doe	49 Leslie St	Toronto	ON	M4M3C3	23676589765	johndoe@gmail.com

Residential_Property

	PropertyID	ListingOfficeID	Street	City	Province	PostalCode
PK & FK constraint	PK	FK				
Data Type	serial	integer	Varchar(55)	Varchar(55)	Char(2)	Char(6)

Nullable	No	No	No	No	No	No
Unique	Yes	No	No	No	No	No
Display format	999	999	111 Xxxxx Xxxx	XXXXX	XX	X1X1X1
Range/valid entries						
Example	125	123	49 Leslie St	Toronto	ON	M4M3C3

Ownership_Percentage

	PropertyID	OwnerID	OwnerPercentage
PK & FK constraint	PK (composite), FK1	PK (composite), FK2	
Data Type	integer	integer	Numeric(5, 2)
Nullable	No	No	No
Uniqueness	No	No	No
Display format	999	999	99.99
Range/valid entries			0.01 to 100.00
Example	125	357	53.35

Listing

	MLSID	PropertyID	Result	InactiveDate	IsSale
PK & FK constraint	PK	FK			
Data Type	Char(8)	integer	Varchar(8)	Date	Bool
Nullable	No	No	Yes	Yes	No
Uniqueness	Yes	No	No	No	No
Display format	Z9999999	999	XXXX	9999-99-99	TRUE/FALSE
Range/valid entries	1 capital letter followed by 7 digits		'Rented', 'Sold', 'Delisted', Null		TRUE/FALSE
Example	C5373827	125	Sold	2021-11-21	TRUE

Sale_Listing

	MLSID	SalesPrice	ClosingDuration	CommissionPercentage
PK & FK constraint	PK, FK			
Data Type	Char(8)	Numeric(13,2)	Numeric(4)	Numeric(5,2)
Nullable	No	No	No	No

Uniqueness	Yes	No	No	No
Display format	Z9999999	9999999	9999	99.99
Range/valid entries	1 capital letter followed by 7 digits	0.01 to 99,999,999,999.99	1 to 9999	0.00 to 100
Example	C5373827	2390000	45	2.50

Assumptions for the above table Sale_Listing:

- Sales Price is positive and less than $100 billion.
- Closing duration is in days and is less than 9999 days
- Commission Percentage is just our part. We don’t include the buyer’s realtors commission.
- Commission Percentage cannot exceed 100%.

Rental_Listing

	MLSID	MonthlyRent	RentalAgreementPeriod	CommissionAmount
PK & FK constraint	PK, FK			
Data Type	Char(8)	Numeric(8,2)	Varchar(9)	Numeric(8,2)
Nullable	No	No	No	No
Uniqueness	Yes	No	No	No
Display format	Z9999999	999.9	99 XXXX	999.9
Range/valid entries	1 capital letter followed by 7 digits	0.01 to 999,999.99	A number followed by ‘Year’,’Month’, or ’Day’ Max 999 month	0.01 to 999,999.99
Example	C5373827	2300	12 Month	1150

Assumptions for the above table Rental_Listing:

- Monthly rent is less than $1 million.
- Commission Amount is less than $1 million.
- Rental Agreement Period is less than 999 months.

Realtek Departmental Reports Query Report

There are 10 sql files attached for departmental reports generation. Please run the 'report_views_run_this_first.sql' first to generate the views required for other queries. Following are the detailed explanation of each file.

1) report_views_run_this_first.sql

This document will generate the nine views required for all the departmental reports. Views are HR_Report , Owned_Office_Detail, Office_Detail, Current_Leased_Office Current_Lessor, Office_Sales_Weekly, Office_Rent_Weekly, Monthly_Total_Commission, and Weekly_Total_Commission. Please run this file before running all the queries for generating reports.

a. HR_Report view:

Shows all employees with their employee IDs, first names, middle initials and last names, their operating manager's first names, middle initials and last names, the office IDs they currently work in and the first names, middle initials and last names of the responsible partner for that office.

b. Owned_Office_Detail view:

Shows all owned offices with their office IDs, office addresses and workforces.

c. Office_Detail view:

Shows all current leased and owned offices with their office IDs, office addresses, workforces and type discriminators.

d. Current_Leased_Office view:

Shows all current leased offices their office IDs, office addresses, workforces and corresponded lease IDs.

e. Current_Lessor view:

Shows all current leased offices with their office IDs, office addresses, corresponded lease IDs, associated lessors' affiliate IDs, first names, middle initials and last names.

f. Office_Sales_Weekly view:

Shows all offices that have at least one sale during the week with their office IDs, office addresses, total sales amount and total sales commissions. This is for Accounting's weekly reports.

Current conditions used for illustrating purpose:

```
AND InactiveDate >= '2021-11-28'
AND InactiveDate < '2021-12-05'
```

If it will be run every week, use following codes on line 76 and line 78:

```
AND InactiveDate >= Current_Date - 7
AND InactiveDate < Current_Date
```

g. Office_Rent_Weekly view:

Shows all offices that have at least one rental during the week with their office IDs, office addresses, total monthly rent and total rental commissions. This is for Accounting's weekly reports.

Current conditions used for illustrating purpose:

```
AND InactiveDate >= '2021-11-28'
AND InactiveDate < '2021-12-05'
```

If it will be run every week, use following codes on line 91 and line 93:

```
AND InactiveDate >= Current_Date - 7
AND InactiveDate < Current_Date
```

h. Monthly_Total_Commission view:

Shows all offices that have a positive commission in a particular month with their office IDs and total commission amounts (including both sales and rental commission). This is for Management's monthly reports.

Current conditions used for illustrating purpose:

```
AND EXTRACT(YEAR FROM InactiveDate) = 2021
AND EXTRACT(MONTH FROM InactiveDate) = 12
```

Change 2021 to the desired year and 12 to the desired month on line 108, 110, 119, & 121:

```
AND EXTRACT(YEAR FROM InactiveDate) =  ****
AND EXTRACT(MONTH FROM InactiveDate) = **
```

i. Weekly_Total_Commission View:

Shows all offices that have a positive commission during the week with their office IDs and total commission amounts (including both sales and rental commission). This is for Accounting's weekly reports.

Current conditions used for illustrating purpose:

```
AND InactiveDate >= '2021-11-28'
AND InactiveDate < '2021-12-05'
```

If it will be run every week, use following codes on line 136, 138, 147 & 149:

```
AND InactiveDate >= Current_Date - 7
AND InactiveDate < Current_Date
```

2) hr_query.sql

Run this every week to get all full-time employees' employee IDs, first names, middle initials and last names, their operating manager's first names, middle initials and last names, the office IDs they currently work in, the first names, middle initials and last names of the responsible partner for that office and the workforce of that office.

Current condition used is order by workforce smallest to largest:

```
ORDER BY WorkForce;
```

If we want to have offices with largest workforce first, use following code on line 12:

```
ORDER BY WorkForce DESC;
```

3) legal_lessors.sql

Run this every month to retrieve all current leased offices with their office IDs, office addresses, corresponded lease IDs, associated lessors' affiliate IDs, first names, middle initials and last names.

4) legal_expiring_leases.sql

Run this every month to retrieve all current leases that are expiring within the next 90 days with their lease IDS, leased office IDs and lease end dates.

5) accounting_rent.sql

Run this every week to retrieve all offices that have at least one rental during the week with their office IDs, office addresses, total monthly rent and total rental commissions.

If we want the bottom 3 rent commissions, use ORDER BY Rent_Commission LIMIT 3; on line 7.

If we want the top 3 rent commissions, use ORDER BY Rent_Commission DESC LIMIT 3; on line 7.

6) accounting_sales.sql

Run this every week to retrieve all offices that have at least one sale during the week with their office IDs, office addresses, total sales amount and total sales commissions.

If we want the bottom 3 sales commissions, use ORDER BY Sales_Commission LIMIT 3; on line 7.

If we want the top 3 sales commissions, use ORDER BY Sales_Commission DESC LIMIT 3; on line 7.

7) accounting_royalty_commission.sql

Run this every week to retrieve all current leased offices that have a positive commission during the week with their office IDs, office addresses and royalty commission amounts.

8) accounting_salary.sql

Run this to retrieve all full-time employees’ employee IDs, first names, middle initials, last names and salaries.

9) management_commission_per_employee.sql

Run this monthly to see which offices are the most efficient. It will retrieve the average commissions earned per employee during a particular month for both corporately owned and leased locations as well as the office IDs and office addresses.

10) management_commission_percentage.sql

Run this monthly to see which offices are the most profitable. It will retrieve the average sales commissions percentage during a particular month for both corporately owned and leased locations as well as the office IDs and office addresses.

Machine Learning and Artificial Intelligence: Future Uses (Competitive Advantage)

In Machine Learning for Databases, Li et al (2021, p. 3190), discuss machine learning optimization as a mechanism for alleviating the following three database-related challenges:

1) NP-hard problems, including index/view selection, partition-key recommendation for offline optimization, query rewrite, and join order selection;
2) Regression problems, such as cost/cardinality estimation, index/view benefit estimation, and query latency prediction; and
3) Prediction problems, like query workload prediction.

These issues are significant because the traditional database requires continuous manual maintenance, which is becoming inefficient with the high-performance requirements of large-scale databases (Li et al., 2021).

Monte Carlo Tree Search (MCTS) for Query Rewriting and Joint Order Selection

Explanation

Query rewriting performs “a number of transformations (independent of the system's physical state) to the original query to produce an equivalent optimized one (Pitoura, 2009, para. 2).” Query rewriting is usually based on a set of heuristic rules or greedy algorithms (Li et al., 2021). Although this process can produce good results, it often fails to utilize high-quality rules and performs slowly with complex queries (Li et al., 2021).

MCTS is an algorithm that searches through a set of available steps for a solution by selecting, expanding, simulating, and updating the tree nodes (Sharma, 2018). It then predicts the optimal path the agent should take to reach the desired result (Sharma, 2018). As a result, MCTS is a powerful tool for the online optimization of NP-hard problems, which require instant feedback (Li et al., 2021). It does so by pre-learning and utilizing a model for online exploration, allowing it to optimize joint order selection and query rewriting (Li et al., 2021).

Benefits for Realtek

Although the competitive landscape for the real estate industry currently indicates strong demand, future growth remains uncertain (PricewaterhouseCooper and Urban Land Institute, 2021). This means that Realtek can expect to continue collecting large volumes of data, however, the nature of it may change frequently. Realtek might collect potential clients' info for marketing purpose. Realtek might want to hire individual realtors and store their information. Realtek might need to store property, office and employee information in another country if they further expand their business. By then, the number of tables & amount of data will be enormous, indicating the potential need for query rewriting and joint order selection. (Blin & Curé, 2014).

Q-Learning for Query Workload Prediction

Explanation

Query workload prediction (or transaction prediction) is the ability to model the workload predictions for database optimization (Li et al., 2021). This assists businesses with resource control, transaction scheduling, and the overall workflow process (Li et al., 2021). In the past, workload predictions relied on rule-based methods that used domain knowledge to identify relevant workload features (Li et al., 2021). The issue here is that each time workload changes, a new model has to be made, making this method inefficient for evolving databases (Li et al., 2021). Q-Learning is an off-policy reinforcement learning algorithm that looks for the optimal action (Violante, 2019). "It's considered off-policy because the q-learning function learns from actions that are outside the current policy, like taking random actions, and therefore a policy isn't needed (Violante, 2019, para. 2)"

Benefits for Realtek

Realtek Inc. comprises multiple departments, each with varying informational needs. As Realtek grows, it might need to analyze its data to find changes and future trends. As such, the ability to accurately predict and schedule transactions would significantly assist with resource control. This could have the added benefit of reducing operational costs, helping Realtek to remain competitive.

Natural Language Interface to Databases (NLIDB)

Explanation

NLIDB utilizes natural language processing, allowing users to formulate queries in natural language. This is significant because it allows individuals to access databases without knowledge of query language (Sangeetha & Hariprasad, 2019).

Benefits for Realtek

The use of NLIDB would provide Realtek with a substantial competitive advantage, as the speed at which users can access the data they require would increase dramatically. This would significantly benefit Realtek for the following reasons:

1) Realtek would rely far less on database administrators to troubleshoot and train database users;
2) Realtek's database users would save time with query processing, increasing the company's overall productivity;
3) This substantial increase in efficiency allows Realtek to expand and grow the company at a faster rate.

None Artificial Intelligence Related Extensions

Data Warehouse

Like mentioned previously, Realtek might need to analyze its data someday to find changes and future trends. Then we need to separate the analytical data from our operational data. Sometime in the future, Realtek might also have customer relationship management (CRM) and enterprise resource planning (ERP) systems in use. They both might store data into potentially different databases. Data warehouse will allow Realtek to pull, integrate and analyze all those data and help on making critical business decisions.

NoSQL Database

The structures of the data that Realtek deals with might also change. Fluctuations between buyer and seller demand would impact the volume of and information of Sale Listings and Property Owners, while changes in rental demand could change the volume of and information of Rental Listings (i.e. geographic trends). At some point, Realtek Inc. might develop property management business like many real estate companies have done. Realtek will might also want to store additional client info like occupation, life event, family relations and additional property info like sq.ft. of the property, year built, historical prices in its database. The current relational database will be able to do the job but might need sacrifice of storage spaces and access times. The structure of many different forms of data is more easily handled and evolved with a NoSQL database. NoSQL databases are often better suited to storing and modeling structured, semi-structured, and unstructured data in one database. NoSQL databases often store data in a form that is similar to the objects used in applications, reducing the need for translation from the form the data is stored into the form the data takes in the code (MongoDB, n.d.).

References

Blin, G., & Curé, O. (Eds.). (2014). *RDF Database Systems: Triples Storage and SPARQL Query Processing*. Elsevier Science.

Li, G., Zhou, X., & Cao, L. (2021). Machine Learning for Databases. *Beijing National Research Center for Information Science and Technology*, *14*(12), 3190-3193.

Pitoura, E. (2009). *Query Rewriting*. Encyclopedia of Database Systems. Retrieved December 11, 2021, from https://link.springer.com/referenceworkentry/10.1007%2F978-0-387-39940-9_863#howtocite

PricewaterhouseCooper & Urban Land Institute. (2021). *Emerging Trends in Real Estate* [Report].

Sangeetha, J., & Hariprasad, R. (2019). An intelligent automatic query generation interface for relational databases using deep learning technique. *International Journal of Speech Technology*, (22), 817-825.

Sharma, S. (2018, August 1). *Monte Carlo Tree Search. MCTS For Every Data Science Enthusiast | by SAGAR SHARMA*. Towards Data Science. Retrieved December 11, 2021, from https://towardsdatascience.com/monte-carlo-tree-search-158a917a8baa

Violante, A. (2019, March 18). *Simple Reinforcement Learning: Q-learning | by Andre Violante*. Towards Data Science. Retrieved December 11, 2021, from https://towardsdatascience.com/simple-reinforcement-learning-q-learning-fcddc4b6fe56

MongoDB. (n.d.). *When to Use NoSQL Databases*. Retrieved December 12, 2021, from https://www.mongodb.com/nosql-explained/when-to-use-nosql