**DB 명명 규칙 가이드**

**DB명명규칙 가이드**

**Ver. 0.8**

**2024. 08.**

**개정이력**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **버전** | **작성일** | **변경내용** | **작성자** | **승인자** |
| V0.1 | 2024-07 | 분석 및 설계단계 완료 | 김기범 |  |
| V0.2 | 2024-08 | 규칙 가이드 보완 및 수정 | 김기범 |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

목 차

[1. 개요 2](#_Toc417921266)

[1.1 목적 3](#_Toc417921267)

[1.2 범위 3](#_Toc417921268)

[2. 논리모델 명명규칙 3](#_Toc417921271)

[2.1 Entity 4](#_Toc417921272)

[2.2 Attribute 5](#_Toc417921279)

[3. 물리모델/DB 명명규칙 8](#_Toc417921298)

[3.1 Table 8](#_Toc417921303)

[3.2 Column 9](#_Toc417921309)

[3.3 Constraint 11](#_Toc417921316)

[3.4 Index 12](#_Toc417921317)

[3.5 Partition 14](#_Toc417921326)

[3.6 Partition Index 15](#_Toc417921327)

[3.7 Synonym 15](#_Toc417921328)

[3.8 Sequence 16](#_Toc417921329)

[3.9 View 16](#_Toc417921330)

[3.10 User Function 17](#_Toc417921331)

[3.11 Strored Procedure 17](#_Toc417921332)

# 개요

본 문서는 “E-Commerce 통합 플랫폼 구축” 프로젝트의 데이터베이스 구성요소들의 가독성을 향상시키고 일관성을 유지하여 개발자의 생산성과 품질을 향상 시키기 위하여 프로젝트 개발, 유지보수를 할 각 오브젝트들과 기타 관련 문서 파일들에 명확한 명칭 부여 기준을 수립함을 목적으로 한다.

## 목적

* 용어 사용의 혼란 방지

표준화된 명칭을 부여함으로써 용어 중복 정의 방지와 불일치를 제거하여 의사 소통을 원활하게 한다.

* 데이터 모델 및 업무 일관성 유지

표준화된 용어를 사용함으로써 모델을 구성하는 오브젝트 명칭의 일관성을 유지할 수 있다.

* 데이터의 정확성 및 품질 수준 확보

표준화를 통해 데이터의 구조와 흐름 등 정합성 검증이 용이하여 품질 수준을 확보할 수 있다.

* 개발 생산성 향상

공통의 용어 정의를 통해 재사용 기회를 높이고 작성 시간을 단축시킴으로써 개발 생산성을 향상

* 데이터 관리의 일관성과 효율성을 높임

## 범위

|  |  |  |
| --- | --- | --- |
| 구성 요소 | 세부 대상 | 내 용 |
| 논리모델 | 엔터티 명  어트리뷰트 명 | * 엔터티/어트리뷰트 명은 표준 단어 사용 |
| 물리모델 | 테이블 명  컬럼 명 | * 통용되는 약어를 사용하고 단어사이에는  underline(\_)으로 구분한다. |
| DB | DB/Schema  DB Object | * 업무영역을 고려하여 정의 * 기본이 되는 오브젝트의 이름을 포함하여 생성되는 오브젝트의 특성별로 이름을 접두/접미 시킴 |

# 논리모델 명명규칙

논리적 데이터 모델은 중복된 데이터가 존재하지 않도록 완전 정규화된 데이터 모델로 작성한다.

## Entity

### 해당 객체(Entity)의 명칭은 개체(Entity)의 목적에 잘 부합되도록 정의

### 개체(Entity)의 명칭은 모호하지 않아야 함

### 개체(Entity)명은 한글로 명명하고, 영문 사용시는 대문자로 표시

### ~관리, ~정보, ~내역, ~현황 등의 명확하지 않은 표현은 사용하지 않아야 함. 단, 부득이하게 ~정보 / ~내역이라는 용어를 사용할 경우가 생기게 되면 ~정보라고 통일해서 사용하는 것을 권장함

예) 고객관리(X), 고객내역(X), 고객정보(X) -> 개인고객(O)

예) 결제정보(X), 결제내역(X) -> 결제(O)

### 고유명사를 사용할 수 있는 개체(Entity)는 고유명사를 그대로 사용

예) 고객, 회원 가맹점 등

### 가급적 고유명사 앞에 수식어를 붙여서 명확화

예) 개인고객, 법인고객, 잠재고객

|  |  |
| --- | --- |
| 적용 구조 | **4 - 30**  **1 - 3**  [1-3] : 서브시스템코드  [4-30] : 엔터티 의미 |
| 명명 규칙 | [1-3] : 서브시스템코드  [4-30] : 엔터티 타입 의미. |
| 작성 사례 | 1. 시스템관리/기준정보 주제영역의 공통코드 엔터티   **공통코드**  **SMP** |

## Attribute

### 표준단어( 주제어 + 수식어1 + 수식어2..) + 도메인

### 단어는 유일하게 작성(동음이의어, 이음동이어 사용금지)

### 표준어 규정 및 한글 맞춤법을 준수

### 표준단어는 가급적 명사형으로 정의하는 것을 원칙.

### 도메인에 해당되는 용어는 반드시 후치로 사용합니다.

예) 금액, 양(량, 비율, 율, 료), 일자(년월, 시간, 개월, 기간), 코드(구분코드, 여부, 유무), 명(성명), 수(건수, 차수), ID(사번) 등

### 논리 데이터 모델은 한글로 기입함을 원칙으로 함

### 한글 명사 간에는 “\_”, “-“ 등을 넣지 않습니다.

### 한글 적용이 불가능한 영문단어일 경우 영문을 사용.

### 영문명의 경우 27자리 이내를 원칙으로 함

### 데이터가 코드 테이블로 관리되는 경우 ~구분코드, ~분류코드, ~유형코드 등으로 사용하고, 코드 테이블로 관리되지 않는 경우 ~구분, ~분류, ~유형 등으로 사용.

### ~사용자, ~등록자 등으로 사용하지 말고, 반드시 ~사용자 ID, ~등록자 ID, ~사용자명, ~등록자명 등처럼 명확하게 사용.

### 일자(YYYYMMDD), 일시(YYYYMMDDHH24MISS) 등은 구분하여 사용.

### ~시간의 의미에 따라 ~시각(tm), ~일자(dt), ~일시(dtime) 등으로 구분하여 명확하게 사용

### ~기간의 의미에 따라 ~시수, ~일수, ~시작일자/종료일자 등으로 구분하여 명확하게 사용

### 한글을 영문명(물리명)으로 전환 시 발음나는 대로 사용하는 것은 지양할 것

### 분해가 더 이상 되지 않는 용어를 등록 사용

#### 예외, 칼럼의 자리수 제한을 초과할 경우 복합어를 등록 사용

#### 속성(Attribute)은 용어 + 용어, 용어는 더 이상 의미적으로 나누어 질 수 없는 단위

### 용어 합성 시 영어 어순이 아니라 한글 어순으로 작성

예) 꽃배달(deliver flower -> deliv\_flwr) => 꽃(flower) + 배달(delivery) -> flwr\_deliv

#### 한글명이 동의어가 아니고 영문은 동일한 경우 영문약어를 상이하게 등록

예) 단독 single -> single, 독신 single -> sngl

### 단위는 미터법을 원칙으로 하되 다른 단위를 사용할 경우 반드시 표시

|  |  |
| --- | --- |
| 적용 구조 | **1-30**  [1-30] : 의미가 전달되는 영문 한글 단어 조합 |
| 명명 규칙 | [1-30] : 표준용어 한글명과 동일하게 사용 |
| 작성 사례 | 부서전화번호 -> 부서(Qualifier) + 전화번호(Domain)  **부서전화번호** |

# 물리모델/DB 명명규칙

## Table

### 서브시스템 코드가 없을 경우 서브시템코드 뒤의 '\_' 문자는 생략하도록 한다.

### 데이터 베이스 제품 종류에 따라서 테이블 명칭의 대소문자가 구분되는 것과 대소문자 구분이 되지 않는 제품이 있다. 따라서 데이터베이스 제품 종류에 투명하며, 일관성 있는 테이블 설계를 위해 모든 테이블 명칭은 대문자만으로 구성한다.

### 테이블 명칭 내에서 각 단어 간의 구분은 '\_'(underline)을 사용하도록 한다.

### 테이블 명칭의 두번째 이후의 단어는 가능한 표준 영어 명사를 사용하도록 하며, 어절의 길이가 길어 짧게 표현할 필요가 있을 경우 이외에는 가급적 약어 혹은 한글 발음 표기를 사용하지 않도록 한다.

### 테이블 명칭의 길이는 가급적 최대 20 자리를 넘지 않도록 한다.

|  |  |
| --- | --- |
| 적용 구조 | **\_**  **1-3**  **5 - 25**  [1-3] : 서브시스템코드  [5-25] : 엔터티 영문명 |
| 명명 규칙 | [1-3] : 서브시스템코드 3자리  [5-25] : 엔터티 영문명  순번식 지정(001, 002)은 사용하지 않음 |
| 작성 사례 | “자재/기준정보” 시스템의 “자재메인” 엔터티의 테이블 ID  **\_**  **???**  **MAT\_MAIN**  서브 시스템코드가 존재하지 않는 "자재메인" 엔터티 테이블 ID  **MAT\_MAIN** |

### 서브시스템 적용 예 - SMP: 시스템, 조직, 사용자 접두 - MC: 상품관련 접두 - MR: 주문 관련 접두 - MS: 정산 관련 접두 - BUDGET: 예산 관련 접두 - LOGI: 물류 관련 접두 - IF: 인터페이스 관련 접두 - T\_CO: 전자입찰 업체 관련 접두 - T\_BI: 전자입찰 업무 관련 접두

## Column

### 데이터 타입은 본래의 데이터 속성(Attribute)에 맞도록 설계하는 것이 원칙.

### 자주 Like 검색이 되는 칼럼의 경우 문자로 설계

### 데이터의 도메인을 명확하게 하기 위하여 CHAR와 VARCHAR를 명확히 구분하여 설계한다.

#### 입력 문자열의 길이가 일정한 경우 가능한 char 타입을 사용하며 varchar 타입은 가급적 사용하지 않도록 한다.

### VARCHAR의 경우 가변길이라고 해서 설계 시 무조건 길이를 4000으로 설계하면 안된다. 2000 byte 이상의 데이터를 저장할 필요가 있을 경우, long 혹은 text 타입을 사용해야 한다. long 혹은 text 타입은 각종 함수(nvl, ltrim), 인덱스(index), like 검색 등을 사용할 수 없다.

### 컬럼 내 데이터가 저장될 때 구분자(delimeter)가 포함되는 경우는 가급적 없도록 설계 시 고려한다. 구분자를 포함한 데이터를 저장하고자 하는 경우에는 하위 테이블(child table)을 별도로 추가한 후 구분자를 이용해 분할 데이터를 하위 테이블에 저장한다.

### 컬럼명 지침

#### 날짜 및 시간 관련 속성명

|  |  |  |  |
| --- | --- | --- | --- |
| 한글명 | 영문명 | 데이터 형식 | 데이터 타입 |
| 일자 | \_DT | 날짜(YYYYMMDD) | VARCHAR(8) |
| 시간(시각) | \_TM | 시각(HHMMSS) | VARCHAR(6) |
| 일시 | \_DTTM | 시스템 데이트(DATE) | DATE |
| 일시분초 | \_DTHMS | 날짜&시각(YYYYMMDDHHMMSS) | VARCHAR(14) |
| 년도 | \_YYYY | 년도(YYYY) | VARCHAR(4) |
| 년월 | \_YYYYMM | 년도&월(YYYYMM) | VARCHAR(6) |
| 월일 | \_MMDD | 월&일(MMDD) | VARCHAR(4) |
| 시분 | \_HM | 시&분(HHMM) | VARCHAR(4) |
| 시간(간격) | \_TM | 두 시각 사이의 간격(시간 단위) | NUMBER(\*\*) |

#### 시스템 추적성 관련 컬럼

|  |  |  |  |
| --- | --- | --- | --- |
| 한글명 | 영문명 | Comment | 데이터 타입 |
| 생성일시 | CREATEDATE | 데이터를 최초 생성한 일시 | DATETIME |
| 생성자ID | CREATORID | 데이터를 최초 생성한 작업자 ID | VARCHAR(10) |
| 변경일시 | UPDATEDATE | 데이터를 최종 변경한 일시 | DATETIME |
| 변경자ID | UPDATERID | 데이터를 최종 변경한 작업자 ID | VARCHAR(10) |

시스템 추적성 관련 칼럼은 생성시 모두 작성하며, 영문명은 개발 시스템의 사용자 테이블 칼럼에 따라 변경 가능

#### 도메인 적용에 따른 칼럼 명칭

|  |  |  |
| --- | --- | --- |
| 유형 | 컬럼명 | 데이터 |
| 코드 | \_CD | 코드 테이블에 정의되어 있는 코드 데이터 |
| 성명 / 명칭 | \_NM | 성명 자리수 통일, 기타 명칭 자릿수 별로 정의 |
| 내역 / 설명 | \_DESC | 자릿수별로 정의 |
| 비고 | \_RMK | 자릿수별로 정의 |
| 여부 | \_YN | Y / N |
| 유부 | \_EXS | 0 / 1 |
| 금액 | \_AMT | 자릿수별로 정의 |
| 건수 | \_CNT | 자릿수별로 정의 |
| 횟수 | \_FREQ | 자릿수별로 정의 |
| 일련번호 | \_SEQ\_NO | 자릿수별로 정의 |
| 아이디 | \_ID | 사용자의 아이디 |
| 상태 | \_ST | 상태값 |
| 순서 | \_ORD | 순서 |
| 합계 | \_SUM | 합계 |
| 날짜 |  | 날짜 및 시간 관련 속성명 참조 |

|  |  |
| --- | --- |
| 적용 구조 | **1-30**  [1-30] : 영문약어의 조합 |
| 명명 규칙 | [1-30] : 표준용어 영문명과 동일하게 사용 |
| 작성 사례 | “현장코드” 컬럼 ID  **SITE\_CD** |

## Constraint

|  |  |
| --- | --- |
| 적용 구조 | **Table 명**  **1-2**  **3-4**  **\_**  [1-2] : 오브젝트 구분  [3-4] : 순번 |
| 명명 규칙 | [1-2] : 오브젝트 구분 2자리  PK - Primary Key  FK - Foreign Key  UK - Unique Key  CK– Check Constraint  [3-4] : FK, UK, CK의 경우 여러 개 있다면 순번 사용  순번은 01~99 까지 사용 가능함  PK에서는 순번을 사용하지 않음 |
| 작성 사례 | 1. 자재 테이블의 Primary Key ID   **???\_MATRL**  **\_**  **PK** |

## Index

### 프로그램 개발 완료 후 각 테이블의 데이터를 추출(select)하는 모든 쿼리를 검사한다. 이중에서 가장 많은 빈도로 호출되는 쿼리를 선택한 후, 쿼리의 응답 속도가 최상이 될 때까지 인덱스를 조정한다.

### 앞서 하나 혹은 두 인덱스에 의해서 다른 쿼리 성능이 떨어지는 현상이 발생하지 않는지 확인한다.

### 가급적 null 값이 허용되고, null 데이터를 많이 포함하는 컬럼은 인덱스 대상에서 제외한다.

### 데이터 값 편차가 적은 칼럼 또한 인덱스 대상에서 제외한다.

### 인덱스 생성으로 인한 성능 향상을 얻을 수 있는 경우는 인덱스를 포함하는 테이블이 데이터가 많거나, 향후 행(row)의 수가 급격히 증가할 것이 예상되는 경우이다.

### 일반적으로 1,000 ~ 10,000건 이하 행을 지니는 테이블에서는 인덱스를 만들어도 성능 향상 효과가 없거나, 최적화기(optimizer)에 의해서 인덱스가 무시될 수도 있다.

### 하나의 테이블에 가급적 3개 이상의 인덱스를 만들지 않아야 한다. 인덱스가 많을 경우, 추가(insert), 수정(update), 삭제(delete)시 쿼리의 반응 속도가 느려지게 되며, 디스크 사용량이 급증한다.

### 

|  |  |
| --- | --- |
| 적용 구조 | **3-4**  **Table 명**  **\_**  **1-3**  [1-3] : 인덱스구분  [3-4] : 일련번호 |
| 명명 규칙 | [1-3] : 인덱스 구분  PK : Primary Key 인덱스  UX : Unique Key 인덱스  IDX : 일반 인덱스  [3-4] : 일련번호는 01~99 까지 사용 가능함  PK에는 일련번호를 사용하지 않음 |
| 작성 사례 | 자재 테이블의 첫 번째 인덱스 ID  **01**  **MAA\_MATRL**  **\_**  **IDX** |

## Partition

|  |  |
| --- | --- |
| 적용 구조 | **1-2**  **\_**  **Table 명**  **3**  **4-10**  [1-2] : ‘PT’  [3] : 주 파티션 종류  [4-10] : 파티션 일련번호 |
| 명명 규칙 | [1-2] : ‘PT’  [3] : 주 파티션종류  H : Hash Partition  L : List Partition  V : Vritual Partiton  R : Range Partition  단, composite partition (Range+Hash, Range+List)은 R만 표기  [4-10] : 파티션 일련번호  Range Partition 인 경우 의미 있는 문자열  (예: 2013, 201304 ) |
| 작성 사례 | 자재 테이블의 1번째 hash partition  **001**  **H**  **PT**  **\_**  **SYA\_MATRL** |

## Partition Index

|  |  |
| --- | --- |
| 적용 구조 | **5-6**  **1-2**  **Table 명**  **\_**  **3-4**  [1-2] : ‘PT’  [3-4] : 인덱스 종류  [5-6] : 일련번호 |
| 명명 규칙 | [1-2] : ‘PT’  [3-4] : 주 파티션종류  PK - Primary Key  UX - Unique Key  IDX - 일반 인덱스(Global Index)  LX - Partition Local Index  [5-6] : 일련번호 |
| 작성 사례 | 자재 테이블의 local index  **01**  **LX**  **PT**  **\_**  **SYA\_MATRL** |

## Synonym

|  |  |
| --- | --- |
| 적용 구조 | **Table명** |
| 명명 규칙 | Table 명과 동일하게 명명 함 |
| 작성 사례 | 자재 테이블 Synonym  **???\_MATRL** |

## Sequence

### 길이는 제한이 없으며 영문 대문자를 사용함

|  |  |
| --- | --- |
| 적용 구조 | **\_**  **1-2**  **3-4**  **Table 명**  [1-2] : ‘SQ’  [3-4] : 일련번호 |
| 명명 규칙 | [1-2] : ‘SQ’  [3-4] : 일련번호 |
| 작성 사례 | 자재 테이블에서 사용하는 시퀀스  **MAA\_MATRL**  **\_**  **SQ**  **01** |

## View

### 길이는 제한이 없으며, 영문 대문자로 작성

### 동일 테이블에 대한 다른 뷰를 생성하는 경우 View 명 뒤에 일련번호를 붙여 구분

|  |  |
| --- | --- |
| 적용 구조 | **1**  **View 명**  **\_**  [1] : ‘V’ |
| 명명 규칙 | [1] : ‘V’ |
| 작성 사례 | 자재 코드 뷰  **V**  **MAA\_MATRL\_CD**  **\_** |

## User Function

### 펑션명 길이는 제한이 없으나 영문 대문자를 사용

### 단어간에는 '\_'(underline)으로 구분함

|  |  |
| --- | --- |
| 적용 구조 | **1**  **함수 명**  **\_**  [1] : ‘FN’ |
| 명명 규칙 | [1] : ‘FN’ |
| 작성 사례 | 자재 코드 함수  **FN**  **MAA\_MATRL\_CD**  **\_** |

## Strored Procedure

### 프로시져 명 길이는 큰 제한이 없으나 영문대문자를 사용(오라클의 Object Name 길이 제한은 있음)

### 기능명은 3개의 단어를 넘지 않도록 함.

### 단어간에는 '\_'(underline)으로 구분함

### 표기방식

|  |  |  |
| --- | --- | --- |
| 기능명 | 명칭 | 설명 |
| INS | INSERT | 단일 테이블의 단순 INSERT 작업인 경우 |
| UPD | UPDATE | 단일 테이블의 단순 UPDATE 작업인 경우 |
| DEL | DELETE | 단일 테이블의 단순 DELETE 작업인 경우 |
| LST | LIST | SELECT 문을 사용하여 조회하는 경우 |
| REG | REGISTER | 트랜잭션을 사용하여 여러 테이블에 입력 작업인 경우 |
| MOD | MODIFY | 트랜잭션을 사용하여 여러 테이블에 수정 작업인 경우 |
| REM | REMOVE | 트랜잭션을 사용하여 여러 테이블에 삭제 작업인 경우 |

|  |  |
| --- | --- |
| 적용 구조 | **프로시져 명**  **\_**  **2-4**  **1**  **\_**  [1] : ‘SP’  [2-4] : 표기방식 |
| 명명 규칙 | [1] : ‘SP’  [2-4] : 표기방식 |
| 작성 사례 | 자재 코드 프로시져  **SP**  **MAA\_MATRL\_CD**  **\_** |