## Розділ 4. Проектування бази даних (БД)

### Розробка концептуальної моделі даних

Етап проектування БД вважається одним із найскладніших етапів створення БД, який не має явно вираженого початку й закінчення. У порівнянні з аналізом вимог до БД або розробленням додатків, проектування БД, на думку багатьох провідних фахівців, є невдало структурованим завданням. Якщо всі етапи створення БД перекриваються один з одним у своїй послідовності, то етап проектування перекривається з усіма іншими етапами.

Етап розробки моделі починається з побудови інформаційної моделі в різних знакових формах, які на завершальній стадії втілюються в комп'ютерну модель. Інформаційна модель, як правило, представляється в тій чи іншій знаковій формі. Таблиця - один із прикладів знакових моделей.

Визначимо таблиці, з яких буде формуватися модель даних. При цьому по кожній таблиці слід відповісти на три питання:

* Для збору, якої інформації потрібна ця таблиця?
* Хто її повинен заповняти?
* У чому полягає основна мета її використання?

Визначаємо кількість таблиць, необхідних для створення повноцінної моделі даних, і представимо коротке обґрунтування такого вибору.

Таблиця **«Користувачі»** створена для обмеження доступу до інформації. Користувачі мають доступ лише до інформації за своїм рівнем допуску.

Таблиця **«Сировина»** створена для миттєвого доступу до інформації про доступну для використання сировину.

Таблиця **«Поставки»** створена для збору інформації про прихід сировини на склад. Заповнюється в ході формування замовлення на поставку сировини.

Таблиця **«Постачальники»** створена для доступу до інформації про постачальників. Заповнюється начальником відділу постачання.

Таблиця **«Заявки на постачання сировини»** створена для відображення інформації щодо замовленої продукції. Заповнюється начальником відділу постачання.

На прикладі двох таблиць розглянемо створення їх полів.

У таблиці «**Сировина**» ми сформували такі поля: код сировини, назва сировини, наявність на складі та закупівельна ціна.

Поле ***«*Код сировини*»*** є первинним ключем таблиці «**Сировина**», за допомогою цього поля ми ідентифікуємо сировину за її кодом.

Поле «**Назва сировини**» показує назву сировини.

Поле «**Наявність на складі**» показує скільки сировини лишилось на складі.

Поле «**Закупівельна ціна**» показує ціну за 1 кг продукції.

У таблиці «**Поставки**» ми сформували такі поля: код сировини, код постачальника, дата прийому, ПІБ приймальника, прийнято, брак.

Поле **«Код сировини»** є первинним ключем таблиці «**Поставки**», за допомогою цього поля ми ідентифікуємо сировину за її кодом.

Поле **«Код постачальника»** відображає код компанії поставника.

Поле **«Дата прийому»** показує дату коли була прийнята сировина від постачальників.

Поле **«ПІБ приймальника»** показує хто приймав сировину в день її поставки на склад.

Поле **«Прийнято»** показує скільки сировини було прийнято.

Поле **«Брак»** показує скільки сировини не пройшло перевірку якості. Відображаємо етапи створення моделі даних в ARIS Express**.**

Повна модель даних створена за допомогою ПЗ «Aris Express»:

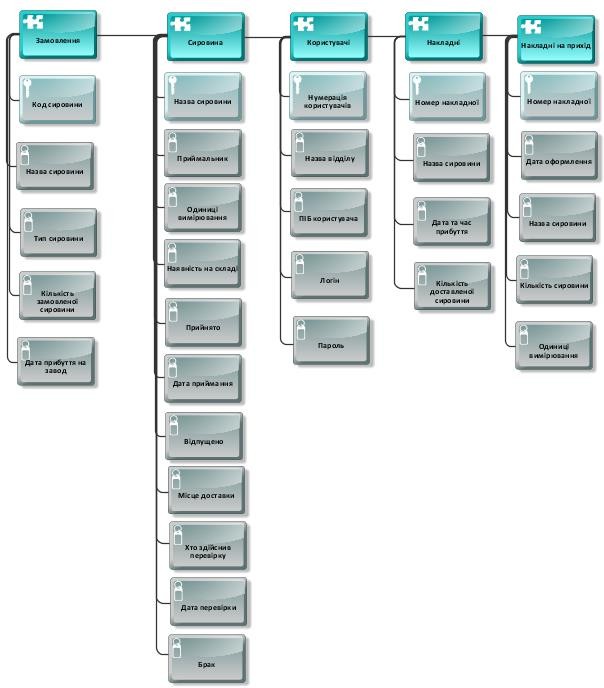


Рис.4.1.1 - Повна концептуальна модель даних

### Розробка структури таблиць бази даних

У процесі створення бази даних в MySQL було з’ясовано, що вона буде складатися з 5-ми таблиць.

Опис таблиць БД з назвою «**winery\_automation**», яка складається з 5 таблиць:

1. users – містить інформацію про користувачів їх логін, пароль та рівень доступу які використовуються для авторизації на сайті.
2. products – містить перелік сировини, що є на складі.
3. inventory\_transactions – містить інформацію про прихід поставок і дати їх здійснення;
4. orders – містить данні про заявки на поставки сировини і дату її здійснення.
5. suppliers – містить дані постачальників

На малюнці нижче представлена структура бази даних, вона відображає взаємозв'язки між таблицями, поля таблиць і їх тип даних, а також властивості - первинні і зовнішні ключі.

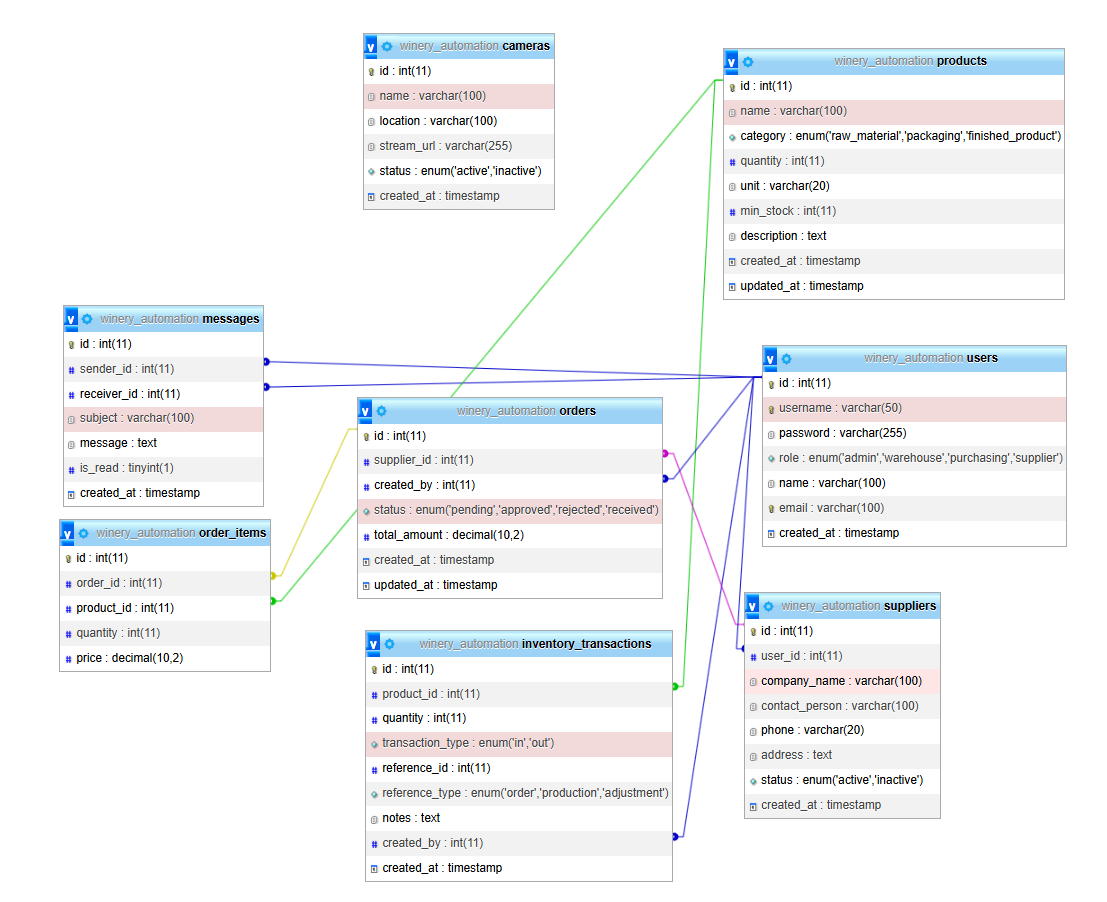


Рис.4.2.1 - Структура бази даних

Коли ми створювали таблиці, ми вказували систему зберігання даних **InnoDB**, це потрібно для правильного функціонування механізмів транзакцій і зовнішніх ключів. Це дозволить нам зберегти цілісність даних при їх змінах.

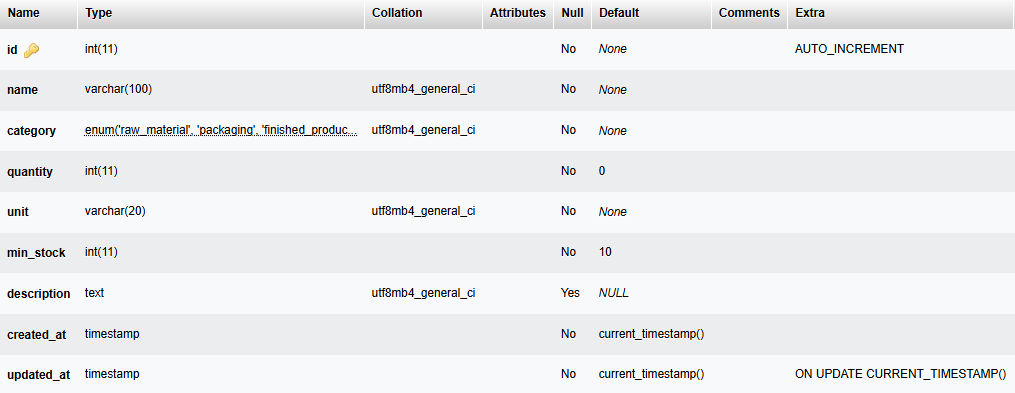
Розглянемо детальніше кожну таблицю

Рис.4.2.2 - Структура таблиці «**products**»

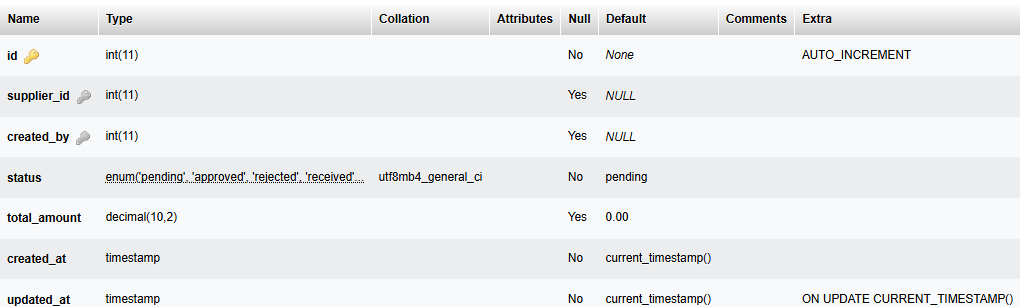


Рис.4.2.3- Структура таблиці «**orders**»

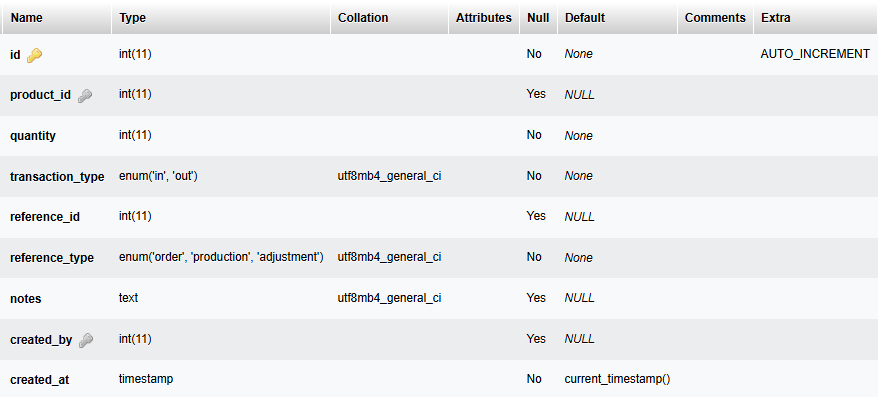


Рис.4.2.4 - Структура таблиці «**inventory\_transactions**»

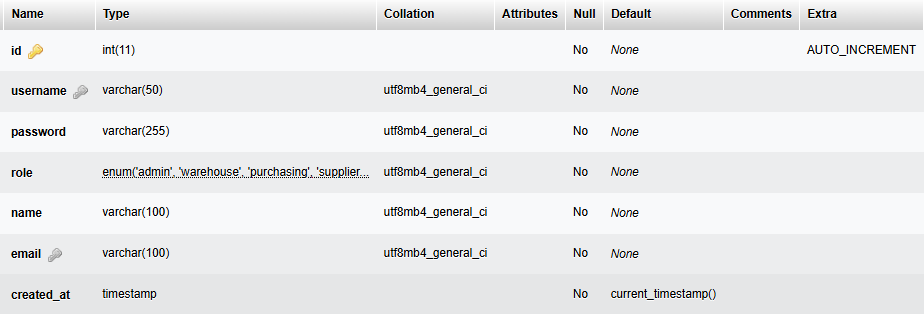


Рис.4.2.5 - Структура таблиці «**users**»

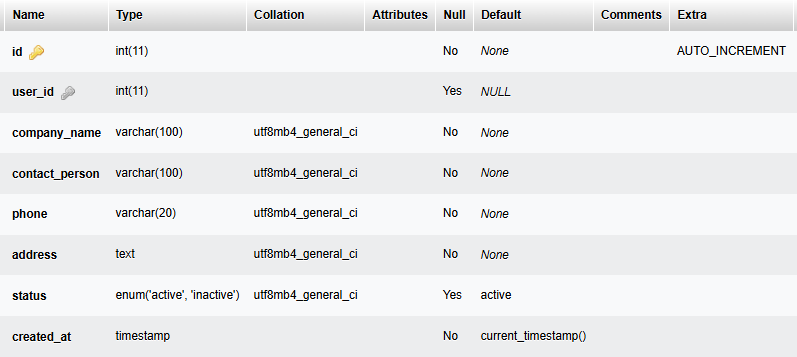


Рис.4.2.6 - Структура таблиці «**suppliers**»